

Eye-in-Hand Manipulation for Remote Handling: Experimental Setup

Longchuan Niu¹, Olli Suominen², Mohammad M. Aref¹, Jouni Mattila¹, Emilio Ruiz³ and Salvador Esque³

¹ Laboratory of Automation and Hydraulics Engineering, Tampere University of Technology, FI-33101, Tampere, Finland

² Laboratory of Signal Processing, Tampere University of Technology, FI-33101, Tampere, Finland

³ Fusion for Energy, Barcelona, Spain

Abstract. A prototype for eye-in-hand manipulation in the context of remote handling in the International Thermonuclear Experimental Reactor (ITER)¹ is presented in this paper. The setup consists of an industrial robot manipulator with a modified open control architecture and equipped with a pair of stereoscopic cameras, a force/torque sensor, and pneumatic tools. It is controlled through a haptic device in a mock-up environment. The industrial robot controller has been replaced by a single industrial PC running Xenomai that has a real-time connection to both the robot controller and another Linux PC running as the controller for the haptic device. The new remote handling control environment enables further development of advanced control schemes for autonomous and semi-autonomous manipulation tasks. This setup benefits from a stereovision system for accurate tracking of the target objects with irregular shapes. The overall environmental setup successfully demonstrates the required robustness and precision that remote handling tasks need.

1. Introduction

Remote handling (RH) is vitally important in the maintenance and operation of the ITER nuclear reactor where human access is impossible. RH tasks are performed indirectly with remotely operated robotic manipulators using man-in-the-loop teleoperation. Vision and force display devices are essential for any teleoperation system to assist operators in manipulation tasks in inaccessible environments [1]. Without haptic feedback [2], peg-in-hole tasks are very difficult to complete when the clearances are very small. Precise and reliable teleoperation could be achieved using marker-based advanced imaging methods, but in the challenging ITER environment, which is under high doses of radiation, any marker object material may suffer from erosion and deformation. Previous work on Computer Aided Teleoperation (CAT) based on 2D template matching and a fixed camera discovered significant limitations of the monocular approach [3]. As a practical solution to cover the above design challenges, in our environment setup, integration of eye-in-hand stereoscopic cameras into the control

¹ The nuclear-fusion reactor, constructed within the ITER project (<http://www.iter.org>).



chain with the specifically developed Iterative Closest Point (ICP) algorithm [4] provides robust and accurate pose estimation and object tracking. This paper is organized as follows. In the next section, the architecture of the experimental setup is introduced by its hardware and software subsystems. Then, section 3 explains the challenges and solutions for integration of stereoscopic visual feedback in such an architecture. Finally, section 4 shows the system performance in practice.

2. System Architecture

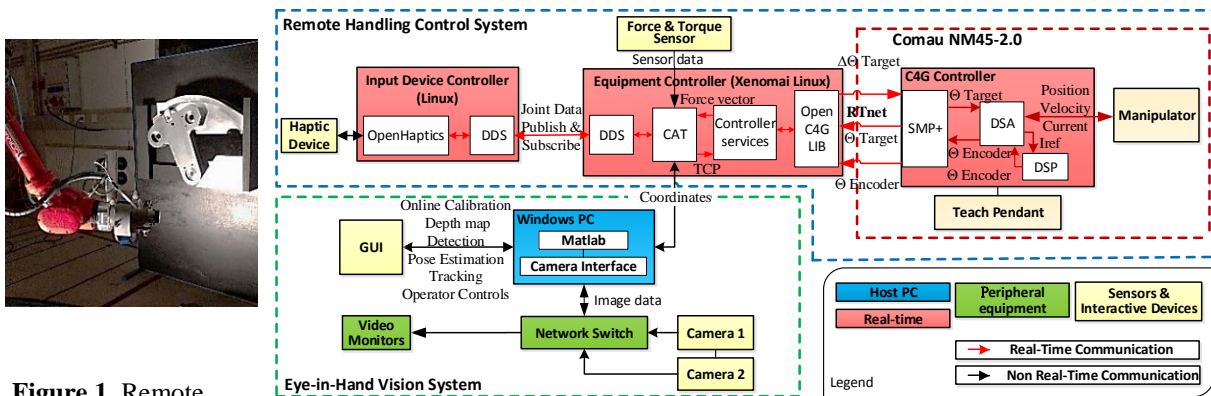


Figure 1. Remote Handling Mock-up Scene.

Figure 2. System architecture.

An experimental setup of the remote handling control system (RHCS) for teleoperation of the Comau SMART NM45-2.0 industrial robot is shown in Figure 1. The system architecture, in Figure 2, comprises the eye-in-hand vision system and RHCS.

2.1. Operating and Development Environments

In the RH architecture, there are a variety of operation environments. Windows environments are flexible and good for user interfaces, however, they do not have characteristics of a real-time operation system, such as determinism. Considering Linux-based systems, Xenomai [5] is a free, POSIX-compatible, real-time kernel extension framework that is easier for programming industrial applications. Similarly, another popular real-time Linux-based system is the Real-Time Application Interface (RTAI) [6]. The performance of both running as dual-kernel schemes is nearly identical [7], provided both use RTnet [8], which is a hard real-time networking framework. The Comau C4G controller provides a software extension called C4G open, which allows for interacting with robots via an external PC (in our case, a Debian Linux PC enhanced by Xenomai) by means of the RTnet protocol stack (Figure 2). The input device controller is another Debian Linux PC installed with OpenHaptics to support haptic devices and OpenSplice data distribution service (DDS) for real-time communication with the Xenomai Linux PC. The host PC for the vision system is a Windows 10 system installed with Matlab and the camera interface for camera access. As explained above, this kind of setup can face heterogeneous subsystems because of the different requirements for each part of the system, especially when it has to be integrated or coordinated with the pre-existing industrial setups.

2.2. Remote Handling Control System

The RHCS is a master-slave structure, where the master input device is the haptic device and the robotic manipulator Comau NM45-2.0 is the slave. The equipment controller connects them in the middle, together with force sensor and the Eye-in-Hand vision system. The Xenomai Linux PC [9] mentioned previously was developed to become an equipment controller featuring the following

functionalities: real-time communication with the Comau C4G controllers, Comau NM45-2.0 low-level control, trajectory generation, Comau NM45-2.0 kinematics, publish and subscribe routines for the robotic joint position data, force sensors reading, and vision data reading and sending. The RH operator uses the eye-in-hand vision application through the Graphical User Interface (GUI) and bilaterally teleoperates the Comau NM45-2.0 manipulator through Phantom Omni, a six-Degree-of-Freedom (DOF) haptic display. The design of the bilateral teleoperation control for the input device controller follows Hooke's law as the virtual impedance:

$$\mathbf{F} = \begin{cases} \mathbf{0} & f(\mathbf{x}) > f(\mathbf{x}_0) \\ \mathbf{K}\mathbf{x} + \mathbf{B}\dot{\mathbf{x}} & f(\mathbf{x}) \leq f(\mathbf{x}_0) \end{cases}$$

where F is the contact force, K is stiffness constant value, B is the damping coefficient along K , \mathbf{x} is the impedance type of penetration vector and \mathbf{x}_0 is a given point in the vicinity of the obstacle. The input device controller is implemented in this impedance-based position-force approach to prevent undesirable movement changes.

2.3. Communication Interfaces

This heterogeneous system requires seamless, time-critical communication. To ensure real-time deterministic communication between the equipment controller and the Comau C4G controller, the RTnet is used to carry the controller specific real-time queues with a 2 ms communication cycle time. Communication between the input device controller and equipment controller was implemented using the OpenSplice DDS middleware at a 1-kHz frequency. Communication between the 3D node and the RHCS is with UDP packets at 1Hz.

3. Integration of Eye-in-Hand Vision System in RH, the 3D Node

The main aim of our project is to discover the potential of stereoscopic imaging for remote handling. The setup of the stereo cameras was done according to the topology design, which takes view angle, stereoscopic precision, and lighting into account. Figure 3 describes the processing stages of the 3D Node module, i.e., the eye-in-hand vision system.

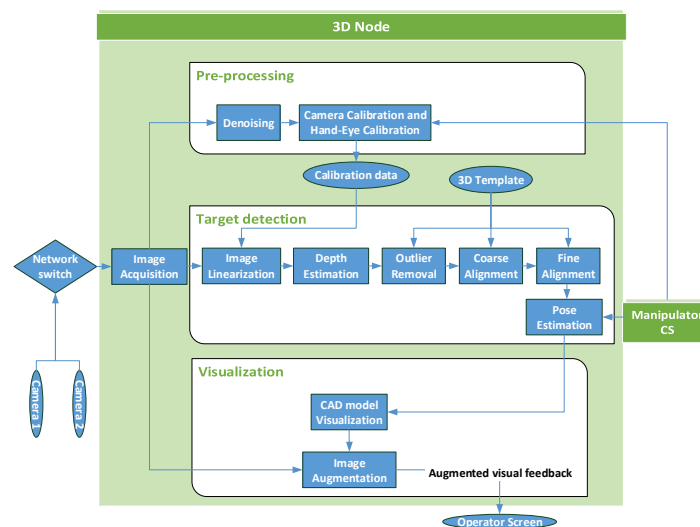


Figure 3. Flowchart of the 3D Node.

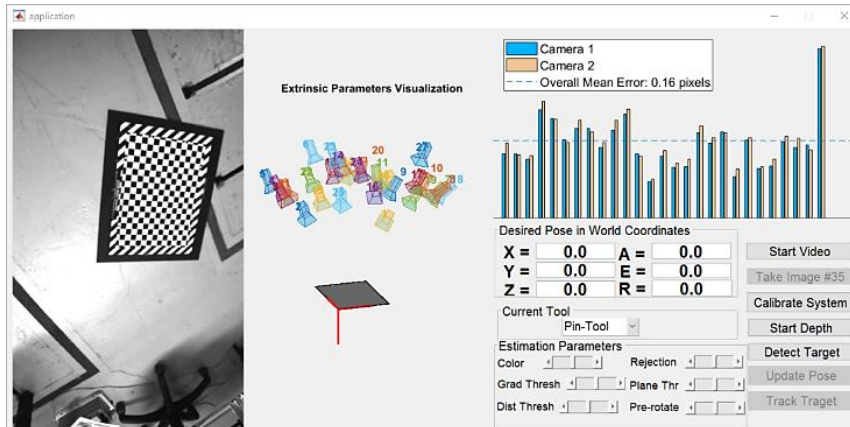


Figure 4. GUI - Camera calibration.

3.1. Implementation of 3D Node

Two GenTL-compliant cameras (Prosilica GE1900) are synchronized using the OUT pin of camera 1 connected with the LINE1 pin of camera 2, automatically triggering both cameras once camera 1 is triggered through the software. This ensures both images have the same content during camera motion. The 3D node acquires images from the stereo cameras through GigE. A setup stage is done outside the operation environment, where the cameras are calibrated using test images of a special calibration pattern (Figure 4). These images are used to extract the intrinsic and extrinsic parameters [10] of the cameras. Meanwhile, the calibration module also performs the hand-eye calibration, referencing the camera's optical center to the tool center point (TCP) of the manipulator. Based on the camera calibration information, the pre-processing stage handles denoising removal of optical distortions (Figure 3). The 3D reconstruction module performs depth estimation (stereo matching) and outlier detection to achieve a clean point cloud of the target scene. This is used for a coarse alignment of the target. The fine alignment step refines the pose estimate by comparing the acquired depth image with the one generated from the known CAD model. This alignment provides the tracking and positioning capabilities while the system is in use. The visualization stage generates the 3D Node GUI with all the user controls and views. In addition, it provides the augmented camera images to the user screen and a visual confirmation of the success of the alignment.

3.2. Mapping of Coordinates

The object pose in camera coordinates has to be transformed into the robot world coordinates, for which the eye-in-hand calibration [11] of extrinsic parameters is essential. With the estimated transform between the model object and the one sensed with the depth-estimation procedure (M_{icp}), one can finally reconstruct a pose of the object relative to the manipulator flange and to the world coordinates (P_{obj}).

$$P_{obj} = R_{robot} X_{handeye} M_{icp}^{-1},$$

where R_{robot} is a pose of a manipulator flange (TCP) and $X_{handeye}$ is the hand-eye matrix, transforming from the camera coordinate space to the manipulator TCP.

The point of interest in the object model, I_{poi} , (i.e., the point where the tool should be inserted), which is manually pre-calibrated in advance, can be re-projected in the camera space using the found fine

alignment. Then, the location of this point relative to the manipulator TCP can be recovered by using the hand-eye relationship matrix, recovered during the eye-in-hand calibration.

$$D_{desired} = T_{tool} X_{handeye}^{-1} R_{robot}^{-1} P_{obj}^{-1} I_{poi},$$

where $D_{desired}$ is the desired location and orientation of the tool prior to insertion (in the world coordinate system), T_{tool} is the tool transformation matrix. The R_{robot} matrix is estimated online using the TCP values from the manipulator.

4. Experimental Results

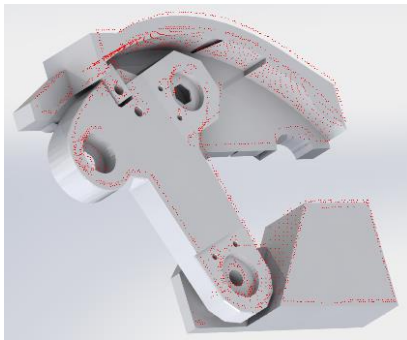


Figure 5. CAD model compared to detected surfaces in the measured point cloud.

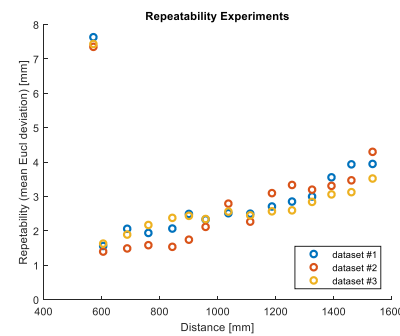
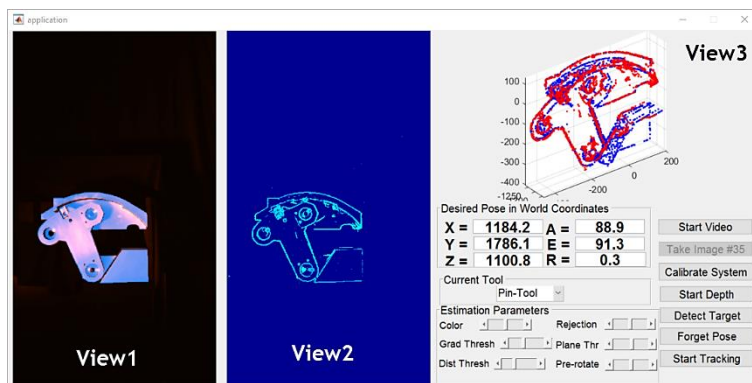


Figure 6. Repeatability (mean deviation) in respect to depth (z -distance).

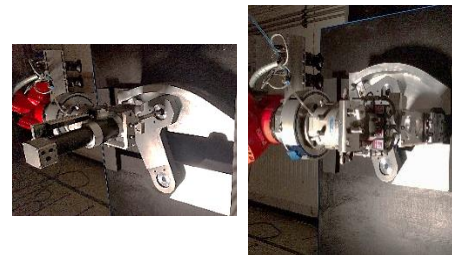
As shown in Figure 5, the target object knuckle can be complicated such that some parts of it can be occluded. Based on the acquired cloud of points (Figure 5, red points), after filtering out the unnecessary parts, it is possible to match with the existing CAD model. This way, the pose of the object is estimated with more accuracy, making it feasible to use its information in the real-time visualization and feedback of the manipulation loop.

One of the important quality aspects for the pose estimation system is its repeatability. That is, a good system can re-measure the same position, taking into account that measurements are independent. We used three sets of experiments. Each of those stereo-images of the knuckle object were taken at different distances ranging from 60 cm to 160 cm. For every experiment, we used a slightly varying orientation (i.e., relative rotation) of a knuckle object, which was approximately the same within all images within the experiment. By measuring the Euclidean distance between the centroid and every other estimate, one can obtain the deviation for a particular stereo-pair. While taking depth, using z -value as a reference variable, we can plot the figure where the horizontal axis is depth (z -distance between camera and the knuckle) and the vertical axis shows the respective deviation values. In Figure 6, when being too close to the target object, camera can no longer observe all the distinctive edges, thus deteriorating overall performance. Within the main operating range, all datasets present good and consistent results. As distance increases, the repeatability error grows but also becomes unstable, which could indicate the existence of an upper limit for the system.



(a) matching (b) depth (c) an estimated pose

Figure 7. (a) the Augmented knuckle completely matched to the real knuckle (b) depth map (c) The point clouds of the knuckle from the CAD model in blue and from the sensed one in red.



(a) the pin tool (b) the jack tool

Figure 8. The scene of the peg-in-hole insertion for the pin tool and the jack tool, where the clearance is only 5mm between the jack tool and the knuckle.

The 3D Node GUI receives user input and provides visual feedback and pose estimation data of the target object, the knuckle. The captured image in Figure 7(a) is augmented with the image of the rendered knuckle at the estimated position. As the captured and rendered images are grayscale, we use color mapping to show two images simultaneously on the same plane. The user can directly see how well images were aligned and verify when the pose estimation was incorrect. This way, even a small angular error can be more noticeable by the user. The 3D node continually processes the live data and displays the estimated pose of the knuckle in the robot manipulator world coordinate, as shown in Figure 7(c). The sampled point clouds of the knuckle from CAD model in blue and the surface point clouds sensed from the real scene in red are shown in view 3, the pose of the detected knuckle is shown at the same time, from which the user can operate the manipulator for a “peg-in-hole” assembly task, as illustrated in Figure 8. The comparative augmented visualization here, significantly improves tool insertion in accuracy and speed.

5. Conclusion

In the current setup environment with the aid of eye-in-hand stereoscopic visual feedback, a RH operator can use the haptic device to accomplish required use cases, such as pickup and drop-off of different tools, and perform peg-in-hole insertion tasks where clearance can be only 5mm. The prototype system has shown to not only increase the speed, usability, and safety of remote handling operation, but also helps the operator to reduce stress. Overall, the experimental setup has demonstrated great reliability, and the system’s precision meets the needs set by the unique ITER operating environment.

6. References

- [1] M. Panzirsch, R. Balachandran, J. Artigas, C. Riecke, M. Ferre and A. Albu-Schaeffer, "Haptic intention augmentation for cooperative teleoperation," 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 5335-5341.
- [2] J. Kofman, Xianghai Wu, T. J. Luu and S. Verma, "Teleoperation of a robot manipulator using a vision-based human-robot interface," in *IEEE Transactions on Industrial Electronics*, vol. 52, no. 5, pp. 1206-1219, Oct. 2005.

- [3] Z. Ziaei, A. Hahto, J. Mattila, M. Siuko, and L. Semeraro, "Real-time markerless augmented reality for remote handling system in bad viewing conditions," *Fusion Engineering and Design*, vol. 86, no. 9, pp. 2033–2038, 2011.
- [4] Rusinkiewicz S, Levoy M. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling*, 2001. Proceedings. Third International Conference on 2001 (pp. 145-152). IEEE.
- [5] Gerum P. Xenomai-Implementing a RTOS emulation framework on GNU/Linux. White Paper, Xenomai. 2004 Apr:1-2.
- [6] Mantegazza P, Dozio EL, Papacharalambous S. RTAI: Real time application interface. *Linux Journal*. 2000 Apr 1;2000(72es):10.
- [7] A. Barbalace, A. Luchetta, G. Manduchi, M. Moro, A. Soppelsa and C. Taliencio, "Performance Comparison of VxWorks, Linux, RTAI, and Xenomai in a Hard Real-Time Application," in *IEEE Transactions on Nuclear Science*, vol. 55, no. 1, pp. 435-439, Feb. 2008.
- [8] J. Kiszka and B. Wagner, "RTnet - a flexible hard real-time networking framework," *2005 IEEE Conference on Emerging Technologies and Factory Automation*, Catania, 2005, pp. 8 pp.-456.
- [9] Alho P., Mattila J. (2013) Real-Time Service-Oriented Architectures: A Data-Centric Implementation for Distributed and Heterogeneous Robotic System. In: Schirner G., Götz M., Rettberg A., Zanella M.C., Rammig F.J. (eds) *Embedded Systems: Design, Analysis and Verification*. IESS 2013. IFIP Advances in Information and Communication Technology, vol 403. Springer, Berlin, Heidelberg
- [10] Forsyth, David, and Jean Ponce, "Computer vision: a modern approach", Upper Saddle River, NJ; London: Prentice Hall, 2011.
- [11] Roger Y Tsai and Reimar K Lenz, "Real time versatile robotics hand/eye calibration using 3d machine vision," in *Robotics and Automation*, 1988. Proceedings., 1988 IEEE International Conference on. IEEE, 1988, pp. 554–561.

7. Acknowledgments

The work leading to this publication has been funded in part by Fusion for Energy and TEKES under Grant F4E-GRT-0689. This publication reflects the views only of the authors, and Fusion for Energy or TEKES cannot be held responsible for any use which may be made of the information contained therein.