Active camera control with obstacle avoidance for remote operations with industrial manipulators: Implementation and experimental results

Magnus Bjerkeng, Aksel A. Transeth, Kristin Y. Pettersen and Erik Kyrkjebø, Sigurd A. Fjerdingen

***Abstract*— Remote controlled robots on offshore oil and gas platforms can potentially reduce costs and improve environment, health and safety issues. A key to successful remote control is to provide onshore operators with a sufficient overview of the processes offshore. To this end, robot manipulators constitute flexible camera platforms, compared to e.g. simple pan-tilt units, for monitoring offshore operations. In this paper, we present a system solution and experimental results for real-time active camera control with obstacle avoidance for industrial manipulators based on weighted pseudoinverse redundancy resolution method. We extend the pseudoinverse solution for joint limit avoidance in combination with obstacle avoidance such that joint limits are graceful avoided. A novel choice of stereographic projection provides robustness with respect global stability and singularities. Implementation issues are addressed and the monitoring approach is experimentally validated on two Kuka KR-16 robot manipulators. Experimental results show that a follower robot with a camera is able to monitor and track a leader robot while simultaneously avoiding collisions. In addition, a robot control system architecture which ensures efficient and safe testing of new Matlab-implemented robot controllers is presented.**

I. Introduction

Offshore oil and gas platforms constitute harsh working conditions and are extremely costly to build, maintain and operate. Environment, Health and Safety (EHS) issues can be improved and costs can be significantly reduced by remote control of offshore operations from onshore. One of the key issues to performing such operations is to provide onshore operators with a sufficient overview of the processes they will control offshore. To this end, robot manipulators can be used for automatic live-video monitoring of operations performed by onshore operators, and this is a topic of this paper.

The view direction of a camera is controlled using two degrees of freedom (DOF) where pan/tilt is a common choice of variables. A standard 6 DOF industrial manipulator is employed in this paper and is thus redundant with respect to the camera task. Hence, the robot is considered to be "task redundant" with respect to camera control using the definition given in [1]. The control of redundant robots has been studied thoroughly in the literature. Redundancy is attractive since the extra degrees of freedom, which do not affect the task performance, may be efficiently used to increase performance with respect to additional criteria such as joint range limits [2], singularity avoidance [3], obstacle avoidance [4] and compliance [5].

M. Bjerkeng and K.Y. Pettersen are with Department of Engineering Cybernetics, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway. {Magnus.Bjerkeng, Kristin.Y.Pettersen}@itk.ntnu.no

A.A. Transeth, Sigurd A. Fjerdingen and E. Kyrkjebø are with SINTEF ICT Applied Cybernetics PO Box 4760 Sluppen, NO-7465 Trondheim, Norway. {Aksel.A.Transeth, Sigurd.Fjerdingen, Erik.Kyrkjebo}@sintef.no

Fig. 1: The floor-mounted robot is connected to stereo vision camera and monitors autonomously operations carried out by the gantry-mounted robot.

The most common strategy to resolve redundancy in real-time applications includes the use of pseudoinverse control and was introduced in [6]. The pseudoinverse is used to resolve redundancy through local optimization of some objective function either at the acceleration level [7] or at the velocity level [8]. A review of velocity level redundancy resolution can be found in [9].

Most previous works consider 3 DOF planar robots or 7 DOF robots as examples of redundant control problems. It is then natural to assume that the reference is given as a full or partial end-effector position and/or orientation trajectory. The controllers are thus developed under the assumption that the control objective is given explicitly as a linear relationship in the task space. For a general task however, no such reference trajectory is available. Care has to be taken in constructing the task parametrization for tasks which are nontrivial functions of the end effector coordinates as this choice directly impacts the closed loop dynamics.

In this paper we develop a new approach for real-time monitoring with obstacle avoidance for industrial manipulators based on the weighted pseudoinverse redundancy resolution method. The main challenge with using the pseudoinverse control method is the requirement of a "well behaved" task parametrization. We demonstrated in [10] that a minimal task representation based on stereographic projection serves as a robust camera error parametrization. this representation has clear advantages over previously proposed methods such as pan-tilt [11], or isometric projection [12] especially with

respect global stability and singularities. We extend the weighted least square pseudoinverse solution for joint limit avoidance in combination with obstacle avoidance such that joint limits are graceful avoided. This is achieved using weighted gradient projected repulsive forces. We extend our results from [10] by demonstrating how to apply the velocity level redundancy resolution method in real-time to an industrial robot system with a limited sensor/actuation interface. The implementation allows the user to set a bound on the joint velocities for added security and ease of tuning for different applications. Experimental results are provided for a leader-follower system which validate the proposed control law. We propose a control system architecture implementation which allows for easy implementation of new robot controllers in Matlab and safe execution of robot motion by the use a System Integrity Module (SIM). The SIM monitors potential robot collisions and inhibits robot commands which may lead to a collision. Moreover, the architecture allows for a seamless switch between control of a simulated and actual industrial robot manipulator.

The paper is organized as follows. Section II presents the laboratory facility and a concept for remote control of offshore operations. Section III contains a description of the control law proposed in this paper, together with an implementation for industrial robot manipulators. The control system architecture for safe execution of robot motions is described in Section IV, and an integrated robot control interface emulator is outlined in Section V. Experimental results are provided in Section VI, and conclusions and suggestions to further work are given in Section VII.

II. Background

In this section a short introduction to the concept of remote inspection and maintenance (I&M) on future normally-unmanned offshore oil platforms is given. Moreover, a lab facility is presented which is used for the implementation and the experimental results presented in this paper.

A. Concept for remote operations

A novel remote I&M concept for offshore oil and gas platforms was presented in [13] as an alternative to traditional offshore platforms. The platform concept is based on separating the work area accessible by human operators, and a closed permanently unmanned area (PUA) that is only serviced by robots.

The PUA's topside location will allow for easier access during I&M operations compared to subsea installations. The remotely operated platform concept is designed on the premise that robots may replace humans for the most important scheduled I&M operations inside the PUA such as gauge readings, valve and lever operations and monitoring leakages, acoustic anomalies and surface conditions [14].

Remote offshore I&M operations pose many significant challenges [15], [13], such as that onshore operators must be able to monitor and control I&M operations with a large range of level of detail. This requires new and versatile active monitoring possibilities which is a topic of this paper.

B. Lab facility for next-generation I&M

A lab facility has been built in Trondheim, Norway, in order to develop, test, and demonstrate solutions for next-generation I&M operations for normally-unmanned oil platforms. A brief overview of the facility is given in the following.

The lab facility consists of a process structure simulating parts of a production process on a real oil-platform and two robot manipulators used for I&M tasks on the process structure using available tools and sensors. See Fig. 2 for a model of parts of the lab facility.

Both robots are standard 6-axes manipulators (Kuka KR-16). One is mounted on a 3-axes gantry. The main tasks of the gantry-mounted robot (GR) is to perform I&M operations on the process equipment. This robot can connect automatically to custom-built tools and sensors such as vibration-measurement sensors, a valve-operating tool, and grippers. The floor-mounted robot (FR) is used for monitoring and assisting the gantry-mounted robot with e.g. a stereo vision camera.

The lab facility can be remotely controlled from any location via the Internet. Live video streams and continuously updated 3D models of the facility provide a remote operator with awareness of the lab operations. The remote operator can initiate high-level commands for automatic I&M routines from a graphical representation of the process equipments, or control the robots with off-the-shelf joysticks either directly or via 3D models. See [13] for further details.

III. Control Strategy

We will in this section describe how the joint reference $\boldsymbol{q}^r(t)$ for the floor-mounted robot is constructed in order to provide camera support in the I&M system. This will be achieved using a leader-follower approach, with the gantry-mounted robot as leader and the floor-mounted robot as follower.

The Kuka KR-16 robot comes with a low level joint position controller, and is given joint angle increments as a reference input. Our control objective is to generate joint angle increments in real-time which achieves camera tracking as well as collision avoidance while respecting inherent joint limitations. The main control task is to generate joint references which achieves camera tracking. The problem of avoiding obstacles, joint limits and controlling the camera view distance will be referred to as sub-tasks.

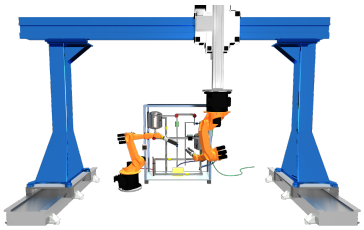


Fig. 2: Robots and process equipment in lab facility.

This section is divided into the following parts; A parametrization of the task space and its kinematics are presented in Section III-A. An overview of the weighted pseudoiverse control method for redundancy resolution is given in Section III-B. A trajectory generating dynamical system is proposed in Section III-C. The sub-task control is given in Section III-D, and a summary of the reference generating system is given in Section III-E. It is described in Section III-F how we use the developed system to generate optimal joint angle references for the follower.

A. The trajectory generating dynamical system

In this section we present the kinematics of a robot manipulator and the parametrization of the task space. The follower is an n-link robot manipulator with joint angles $\boldsymbol{q} \in \mathbb{R}^n$. The forward kinematics of the follower is given by [16]

$$\mathbf{T}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{R}(\boldsymbol{q}) & \boldsymbol{x}(\boldsymbol{q}) \\ \mathbf{0} & 1 \end{bmatrix} \tag{1}$$

where $\mathbf{R}(\boldsymbol{q}) = [\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3] \in SO(3)$ is a rotation matrix composed of unit vectors $\boldsymbol{e}_i$ describing the orientation of the end-effector, and $\boldsymbol{x} \in \mathbb{R}^3$ denotes its position. We will without loss of generality assume that the forward kinematics is constructed such that $\boldsymbol{x}$ is the camera lens position and $\boldsymbol{e}_3$ points in the view direction of the camera. The manipulator Jacobian $\mathbf{J} \in \mathbb{R}^{6\times n}$ maps the joint velocities $\dot{\boldsymbol{q}}$ to the linear velocity $\dot{\boldsymbol{x}}$, and angular velocity $\boldsymbol{\omega}$, of the end-effector:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \boldsymbol{\omega} \end{bmatrix} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2}$$

The objective is to generate a joint reference trajectory $\boldsymbol{q}^r(t)$ for the follower, such that if $\boldsymbol{q}(t) = \boldsymbol{q}^r(t)$ then the follower directs its camera towards a point in space described by the trajectory $\boldsymbol{p}(t) \in \mathbb{R}^3$. We will denote variables pertaining to the follower reference system by a superscript r, to emphasize that we are talking about the reference dynamics, and not the complete follower robot dynamics which is unknown. It was shown in [10] that a stereographic projection of the vector $\boldsymbol{y} = \mathbf{R}(\boldsymbol{q}^\mathbf{r})^T(\boldsymbol{p} - \boldsymbol{x}^r)/\|\boldsymbol{p} - \boldsymbol{x}^r\| \in \mathbb{R}^2$ serves as a well behaved parametrization of the camera tracking error. The stereographic projection used is given by

$$\boldsymbol{\Psi}(\boldsymbol{q}^r) = \begin{bmatrix} \frac{y_1}{y_3+1} & \frac{y_2}{y_3+1} \end{bmatrix}^T, \tag{3}$$

where $\boldsymbol{\Psi}$ is the camera direction error. The error is zero when the camera is directed towards $\boldsymbol{p}(t)$, and undefined when the camera is pointing in exactly the opposite direction. The camera direction error dynamics is given by

$$\dot{\boldsymbol{\Psi}} = \mathbf{J}_t(\boldsymbol{q}^r, \boldsymbol{p})\dot{\boldsymbol{q}^r} + \mathbf{P}(\boldsymbol{q}^r, \boldsymbol{p})\dot{\boldsymbol{p}}, \tag{4}$$

by direct differentiation. The matrices $\mathbf{J}_t = \frac{\partial \boldsymbol{\Psi}}{\partial \boldsymbol{q}^r} \in \mathbb{R}^{2\times n}$ and $\mathbf{P} = \frac{\partial \boldsymbol{\Psi}}{\partial \boldsymbol{p}} \in \mathbb{R}^{2\times 3}$ are derived in closed form in [10]. The matrices $\mathbf{J}_t$ and $\mathbf{P}$ maps the joints velocities to the change in camera error. We will refer to $\mathbf{J}_t$ as the *task Jacobian*. We will assume that the reference point $\boldsymbol{p}$ is changing slowly over a time step, see Section III-F, such that $\dot{\boldsymbol{p}} = \mathbf{0}$ is a good approximation. The camera direction error dynamics then becomes

$$\dot{\boldsymbol{\Psi}} = \mathbf{J}_t(\boldsymbol{q}^r, \boldsymbol{p})\dot{\boldsymbol{q}^r}. \tag{5}$$

This relationship together with $\boldsymbol{\Psi}$ is what we need in order to construct the pseudoinverse redundancy resolution dynamical system.

B. The weighted pseudoinverse redundancy resolution method

We present a short overview of the velocity level weighted pseudoinverse redundancy resolution method such that we may view our extension in the proper context. A thorough overview may be found in [9]. We consider the weighted least squares solution since it allows for greater design freedom than the unweighted least squares solution. The weighted least squares solution to (5) for $\dot{\boldsymbol{q}^r}$ which minimizes $\dot{\boldsymbol{q}^r}^T\mathbf{W}\dot{\boldsymbol{q}^r}$ with the weighing matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$ is given by

$$\dot{\boldsymbol{q}^r} = \mathbf{J}_w^+\dot{\boldsymbol{\Psi}} + (\mathbf{I}_{n\times n} - \mathbf{J}_w^+\mathbf{J}_t^T)\boldsymbol{F}. \tag{6}$$

The $n \times n$ identity matrix is denoted $\mathbf{I}_{n\times n}$, $\boldsymbol{F} \in \mathbb{R}^n$ is arbitrary and $\mathbf{J}_w^+ \in \mathbb{R}^{n\times 2}$ is the weighted pseudoinverse of $\mathbf{J}_t$ given by

$$\mathbf{J}_w^+ = \mathbf{W}^{-1}\mathbf{J}_t^T(\mathbf{J}_t\mathbf{W}^{-1}\mathbf{J}_t^T)^{-1}, \tag{7}$$

which satisfies $\mathbf{J}_t\mathbf{J}_w^+ = \mathbf{I}_{2\times 2}$. If we set $\dot{\boldsymbol{\Psi}} = -k_p\boldsymbol{\Psi}$ for some $k_p > 0$ in (6) then we get the system

$$\dot{\boldsymbol{q}^r} = -k_p\mathbf{J}_w^+\boldsymbol{\Psi} + (\mathbf{I}_{n\times n} - \mathbf{J}_w^+\mathbf{J}_t^T)\boldsymbol{F}. \tag{8}$$

This is called the weighted pseudoinverse redundancy resolution method since $\boldsymbol{\Psi} = \mathbf{0}$ is now stabilized. We observe this by premultiplying (8) by $\mathbf{J}_t$ and using (5); we are then left with $\dot{\boldsymbol{\Psi}} = -k_p\boldsymbol{\Psi}$ as $\boldsymbol{F}$ cancels out. The system (8) hence has $\boldsymbol{\Psi} = \mathbf{0}$ as a globally exponentially stable equilibrium point for all $k_p > 0$ assuming that $\text{rank}\{\mathbf{J}_t\} = 2$.

The vector field $\boldsymbol{F}$ may be used to optimize solutions of (8) with respect to sub-tasks without affecting the stability of $\boldsymbol{\Psi} = \mathbf{0}$. Let a scalar function $U(t, \boldsymbol{q}^r)$ denote how "good", e.g. with respect to the sub-task, a solution to (8) is, and set $\boldsymbol{F} \in \mathbb{R}^n$ as the steepest descent direction $\boldsymbol{F}(\boldsymbol{q}^r) = -(\nabla U(t, \boldsymbol{q}^r))^T$, then solutions to (8) will minimize U in the task nullspace. This technique is called the *Gradient projection method*. Motions induced by $\boldsymbol{F}$ are called *self motion* or *nullspace motion*, as they span the nullspace of $\mathbf{J}_t$.

It was proposed in [17] to use $\mathbf{W}$ to avoid joint limits. The weighing matrix was constructed as a diagonal positive definite matrix where the diagonal elements tends to infinity as a joint reaches its limit. This ensures that joint velocities tends to zero close to a limit assuming that $\boldsymbol{F} = \mathbf{0}$. A weight is set to 1 if the joint velocity increases the distance from a limit such that only undesirable joint velocities are reduced.

C. The weighted gradient solution

A drawback with the solution (8) becomes apparent if we consider the joint dynamics when a joint is close to its limit when $\boldsymbol{F} \neq \mathbf{0}$. If the i'th joint is close to its minimum q_{im}, i.e. $q_i \approx q_{im}$ such that $\mathbf{W}_{i,i}^{-1} \approx 0$, then we see from (8) that $\dot{q}_i = F_i$. This means that F_i needs to be dominated by a joint limit repulsive term in order to guarantee joint limit avoidance. However if this such a term is included, undesirable oscillatory behavior may ensue [17], and the system (8) becomes stiff with respect to numerical integration. We propose to solve this problem by weighing the vector field $\boldsymbol{F}$ by setting $\boldsymbol{F}_w = \mathbf{W}^{-1}\boldsymbol{F}$, and using $\boldsymbol{F}_w$ instead of $\boldsymbol{F}$ in (8). This results in non-oscillatory joint motion when joints are close to their limit as the joint velocity tends to zero rather than arbitrary high values. It is required that $\mathbf{W}_{i,i}^{-1}\boldsymbol{F}_i = 0$ for all joint angles sufficiently close to their limit. This is imposed by choosing the order of $\mathbf{W}_{i,i}$ higher than that of $\boldsymbol{F}$. The weighted gradient least norm solution we propose is given by

$$\dot{\boldsymbol{q}^r} = -k_p \mathbf{J}_w^+ \boldsymbol{\Psi}(\boldsymbol{q}^r) + (\mathbf{I} - \mathbf{J}_w^+ \mathbf{J}_t)\mathbf{W}^{-1}\boldsymbol{F}. \tag{9}$$

The system (9) has $\boldsymbol{\Psi} = \mathbf{0}$ as a globally exponentially stable equilibrium point for all $k_p > 0$ assuming that rank$\{\mathbf{J}_t\} = 2$. We choose a diagonal positive definite weighing matrix inspired by [17] which is a function of the joint angles $\boldsymbol{q}^r$ and their upper and lower limits $\boldsymbol{q}_M^r$ and $\boldsymbol{q}_m^r$,

$$\mathbf{W}_{i,i}^{-1} = \frac{1}{1 + H_i(q_i^r)^2} \tag{10}$$

where

$$H_i(q_i^r) = \frac{(q_{iM}^r - q_{im}^r)^2(2q_i^r - q_{iM}^r - q_{im}^r)}{(q_i^r - q_{iM}^r)^2(q_i^r - q_{im}^r)^2}. \tag{11}$$

The diagonal elements of $\mathbf{W}^{-1}$ tends to zero as a joint approaches its limit, and 1 if a joint is in the middle of its range.

D. The sub-task control

In this section we present a nullspace control input $\boldsymbol{F}$ which is used for optimization of the sub-tasks obstacle avoidance, view distance control and camera roll control. We have used the common repulsive force suggested in [18] for obstacle avoidance,

$$\boldsymbol{F}_{\text{rep}}^i = \mathbf{J}_{\boldsymbol{o}_i}^T \left(\frac{1}{d(\boldsymbol{q}^r)} - \frac{1}{\rho_0} \right) \frac{1}{d^3(\boldsymbol{q}^r)} (\boldsymbol{o}_i(\boldsymbol{q}^r) - \boldsymbol{b}_i) \tag{12}$$

which is set to zero for collision distances $d(\boldsymbol{q}^r)$ larger than $\rho_0 > 0$. Bounding spheres have been used in the distance estimation to optimize the calculation speed. The point $\boldsymbol{o}_i(\boldsymbol{q}^r) \in \mathbb{R}^3$ is the center of a bounding sphere on the follower robot reference and $\boldsymbol{b}_i \in \mathbb{R}^3$ is a the center of a bounding sphere on the leader. The collision distance is given by $d = \|\boldsymbol{o}_i - \boldsymbol{b}_i\| - r_o - r_b$, where r_o and r_b are the radii of the bounding spheres. The matrix $\mathbf{J}_{\boldsymbol{o}_i} \in \mathbb{R}^{3\times 6}$ is the Jacobian $\frac{\partial \boldsymbol{o}_i}{\partial \boldsymbol{q}^r}$. We have used one such repulsive force for each joint segment of the follower, checking collision against the leader robot and the floor. The resulting repulsive force $\boldsymbol{F}_{\text{rep}}$ is the sum of all these forces.

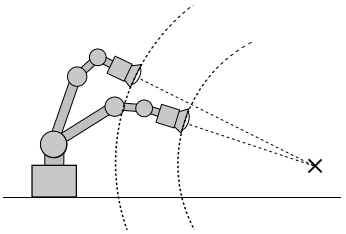


Fig. 3: Conceptual illustration of self motion with respect to the camera view task.

Collision avoidance is achieved with this repulsive force if it is locally feasible in the nullspace of the task Jacobian. If the collision distance drops below a set minimum distance then collision avoidance might not be feasible in the nullspace. The camera task is then temporarily relaxed, and we switch to a pure obstacle avoidance controller

$$\dot{\boldsymbol{q}^r} = \mathbf{W}^{-1}\boldsymbol{F}_{\text{rep}}. \tag{13}$$

This is implemented as an extra security feature for acute collision situations.

The operator may set the desired view distance $d_r = \|\boldsymbol{p} - \boldsymbol{q}^r\|$ as a zoom feature. The view distance is controlled in the task nullspace. It is not included in the main task $\boldsymbol{\Psi}$ in order to maximize the dimension of the task nullspace. If the view distance was included in the main task $\boldsymbol{\Psi}$, then the dimension of the nullspace would decrease by one, and successful nullspace obstacle avoidance would be less likely (see Fig. 3). The desired view distance is controlled by a proportional gain along the desired view vector $\boldsymbol{p} - \boldsymbol{x}^r$,

$$\boldsymbol{F}_{\text{d}} = -k\mathbf{J}_v^T \frac{\boldsymbol{p} - \boldsymbol{x}}{\|\boldsymbol{p} - \boldsymbol{x}\|}(d_r - \|\boldsymbol{x} - \boldsymbol{p}\|), \tag{14}$$

where $k > 0$ is a scalar gain. The matrix $\mathbf{J}_v$ consists of the first three rows of the manipulator Jacobian (2). The desired distance gain is proportional to the distance error and is zero if $\|\boldsymbol{x} - \boldsymbol{p}^r\| = d_r$. The view distance control input is imposed as a proportional gain in order to give it a low priority. The collision avoidance force $\boldsymbol{F}_{\text{rep}}$ will dominate the view distance control input since $\boldsymbol{F}_{\text{rep}}$ is unbounded. This gives the follower increased maneuverability with respect to obstacle avoidance. The view distance can be considered to be a *soft constraint* as opposed to collision avoidance and camera view control which are hard constraints.

The camera roll angle is controlled to ensure that the captured image has a consistent up-direction, and is for instance not upside down. The angle q_6 rotates the camera around its view axis, and does not affect the camera view angle error. This is a common joint configuration for industrial manipulators. The camera roll is controlled directly using q_6^r in the nullspace with the gain

$$\dot{q}_6^r = -k_c \text{Atan2}(R_{3,2}(\boldsymbol{q}^r), R_{3,1}(\boldsymbol{q}^r)). \tag{15}$$

The velocity of q_6^r is proportional to the angle between the cameras up-vector $\boldsymbol{e}_1^r$, and the workspace up vector $[0, 0, 1]^T$

in the plane normal to $\boldsymbol{e}_3^r$. The control law (15) produces an exponentially stable camera roll angle for all $k_c > 0$. If the camera points straight up or down, then (15) is not defined, this ambiguity is fixed by setting $\dot{q}_6^r = 0$ close to these configurations.

E. Overview of the control features

In this section we present an overview of the key features of the trajectories generated by the system (9). A joint trajectory $\boldsymbol{q}^r(t)$ which is a solution to (9) produces an exponentially stable camera direction error for the follower robot such that an attached camera will be directed towards a given reference $\boldsymbol{p}(t)$. The reference $\boldsymbol{p}(t)$ is given in the work-space and will typically be the position of the end effector of the leader robot. No inverse kinematics is required. Only current time information is used, which results in a computationally light controller suitable for real-time implementation. The resulting trajectory is locally optimal with respect to obstacle avoidance and camera control. Kinematic singularities will normally have no adverse effects on the solutions [10]. Joint limits are avoided without undesirable oscillations. The robot specific information used for the follower is the forward kinematics, the manipulator Jacobian for various points on the robot and the location of its joints. The system is easily portable to other robot configurations as the required information is easily constructible given the geometry of a robot.

F. Construction of optimal angle increments

We will in this section describe how the desired follower robot joint increment $\delta\boldsymbol{q}$ is generated at each time-step from (9). The system (9) provides locally optimal joint velocities, and must be numerically integrated on-line in order to produce an angle increment since Kuka KR-16 robots do not allow for direct control of joint angle velocities. The SIM imposes bounds on the maximum allowed angle increment for safety reasons. The implementation of the numerical integrator must respect this constraint as well as produce an angle increment which is sufficiently large. Measurements of the position of the leader robot $\boldsymbol{p}[t_i]$, the joint angles of the follower robot $\boldsymbol{q}[t_i]$ and the desired view distance $d_r[t_i]$ are updated at each time-step t_i. The system (9)

$$\dot{\boldsymbol{q}}^r = \boldsymbol{f}(\boldsymbol{q}^r, \boldsymbol{p}[t_i], d_r) \tag{16}$$

with the initial condition

$$\boldsymbol{q}^r(0) = \boldsymbol{q}[t_i], \tag{17}$$

is numerically integrated on-line. The explicit 4th order Runge-Kutta method was used. We integrate (16) numerically until we exceed the maximum joint increment or until we run out of CPU time. This is done such that the generated joint increment will be the largest possible step towards the optimum. The maximum allowed CPU time is set below the sample time, such that the integration is reset when a new measurement is available. An illustration of the scheme is seen in Fig. 4. With this approach we know that a locally optimal joint angle increment which is as large as possible will be produced.

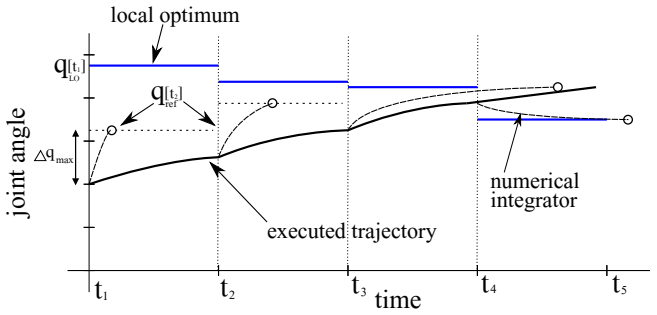


Fig. 4: The implemented on-line optimization scheme. The reference system is integrated numerically stopping when the maximum allowed joint increment is reached or when the next sample is available. The generated angle increment is used as a reference in the low level position controller at the next time step.

IV. Control system architecture implementation

In this section we outline a robot control system architecture for safe testing and execution of coordinated robot motions. This system renders a Matlab implementation of the control methodology presented in Section III possible for control of industrial robot manipulators.

The control system architecture facilitate both the possibility to test new robot controllers in a robot simulator and visualization tool (see Fig. 5), and to seamlessly switch to control of two actual robot manipulators (see Fig. 6). All motion controllers can implemented in Matlab which facilitate easy testing of new controllers. A *High-Level Interface (HLI)* handles communication via TCP/IP between Matlab and a *System Integrity Module (SIM)* for a specific robot. The High-Level Interface also ensures that the control signals for each robot are only transmitted at a pre-defined sample rate. All motion control signal from Matlab must be sent as increments in joint angles $\delta\boldsymbol{q}$ to the SIM due to limitations in the Kuka robot control interface, and current robot joint angles $\boldsymbol{q}$ are sent back to Matlab. The SIM ensures robot integrity by enforcing hard constraints with max-min limits on the robot joint angles and by avoiding robot-robot and robot-environment collisions. Collisions are avoided through the use of a model-based collision-checking software called *CRASH* in a modular approach as illustrated in Fig. 6. CRASH monitors the shortest distance between the robots and their immediate environment. The SIM uses this information to stop a robot if it is too close to its environment. One of the advantages of relaying all robot control signals through the SIM now becomes apparent; The motion control algorithms implemented in Matlab can be tested on the actual robots without having to worry about robot collisions. A program called *Robot Server* is used to relay all sensor and control signals to and from the robots. In particular, Robot Server communicates with the Kuka robot control PCs via a Kuka Robot Sensor Interface (RSI). The RSI is limited to accept only positional increments (e.g., $\delta\boldsymbol{q}$) for motion control (i.e., velocities and accelerations are not

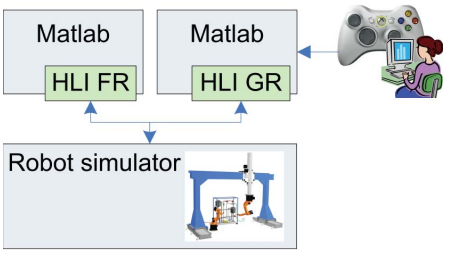


Fig. 5: Conceptual illustration of system architecture for robot control simulation. Matlab transmits reference joint space corrections $\delta \boldsymbol{q}$ and receives current robot joint coordinates $\boldsymbol{q}$ from the simulator for both the gantry robot (GR) and floor robot (FR).

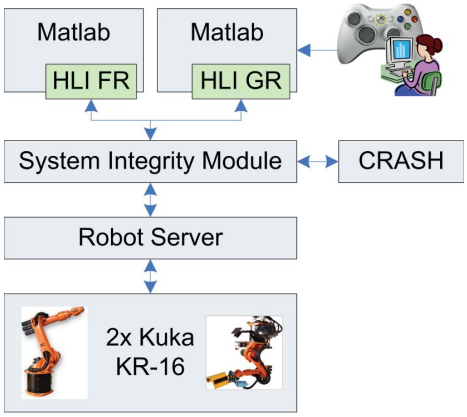


Fig. 6: Conceptual illustration of system architecture for robot control. Matlab transmits reference joint space corrections $\delta \boldsymbol{q}$ and receives current robot joint coordinates $\boldsymbol{q}$ for both the gantry robot (GR) and floor robot (FR). A Java program called a high-level interface (HLI) handles communications to and from Matlab via TCP/IP.

possible to control directly for each time step).

V. Robot visualization and control interface emulator

In this section a short overview of a robot visualization and control interface emulator software tools is given (denoted "Robot Simulator" in Fig. 5). The simulation and visualization tool provides 3D visualization of CAD models of the robots in our lab facility (see Fig. 7). In addition, the camera view from the floor mounted robot is also displayed. A visualization software to display 3D CAD models has been developed by SINTEF Marintek AS. This software together with a new TCP/IP-based interface for updating the positions and orientations of the models (developed for this paper) are used for visualization of CAD-models of the floor robot and the gantry robot. The TCP/IP-interface is identical to the interface provided by the System Integrity Module for control of the actual laboratory robots (see Fig. 6). To this end, new control algorithms written in Matlab can be tested in the simulator, before seamlessly moving on to being

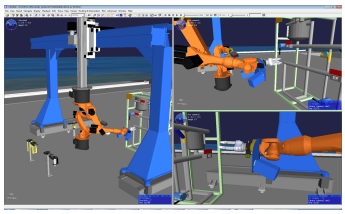


Fig. 7: Floor robot and gantry robot visualized in SimVis. The bottom right image displays the floor robot camera view.

employed on the actual robots. The simulator is limited to describing the robot kinematics and not robot dynamics since input torques and velocities are not possible to send as commands to the industrial robot manipulators.

VI. Experimental Results

In this section the experimental setup is described. Moreover, four separate experiments were performed and these are presented and analyzed in order to highlight different aspects of the closed loop robot controller performance. The numerical integration of (9) was implemented in Matlab (R2009b), and was ported to C in order to minimize its runtime and overhead.

A. Experimental setup and overview

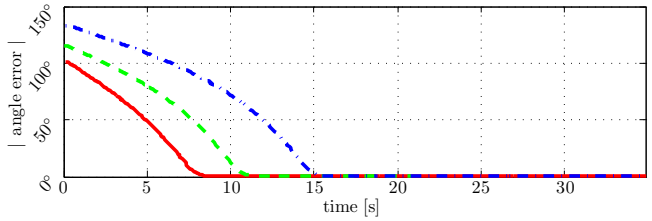
The experimental setup consists of the floor robot (FR) and gantry robot (GR) described in Section II-B and illustrated in Fig. 2. An operator can perform various joystick-controlled operations with the GR such as using a gripper. The FR holds a stereo vision camera as illustrated in Fig. 1 and is controlled by the algorithms controller developed in this paper and [10] in order to automatically monitor the GR end-effector. This provides the operator with a clear view of the operation performed with the GR. The desired relative distance between the FR and the GR is operator controlled in order to provide various levels of "zoom" with the FR-mounted camera.

B. Approach transient

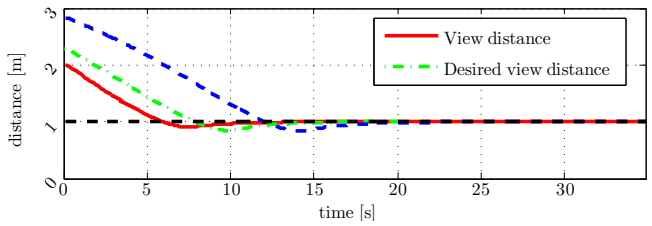
The first experiment shows the transient of the follower robot following an initial condition with a large error. Three separate experiments were conducted. The leader robot is static throughout the run. The camera angle error and the view distance is shown in Fig. 1 and Fig. 8. The camera angle error is monotonically decreasing since it is an exponentially stable equilibrium of (9). The view distance error does not have this exponential characteristic since it is controlled by a proportional gain in the task-nullspace.

C. Camera tracking

This experiment shows the tracking performance as the leader robot is moving around. The focus point $\boldsymbol{p}$ is set to be the end effector position of the leader. The initial error is small. The leader robot is freely controlled on-line with inputs given by an Xbox gamepad such that only delayed position measurements of the leader can be used. Plots of



(a) The absolute angle error $\text{acos}(\boldsymbol{e}_3^T(\boldsymbol{x}-\boldsymbol{p})/\|\boldsymbol{x}-\boldsymbol{p}\|)$ during three separate transient approaches



(b) The viewing distance $\|\boldsymbol{x}-\boldsymbol{p}\|$ during transient approach.

Fig. 8: Data from three transient approach experiments where the focus point is fixed and the initial error is large.

the experiment data is shown in Fig. 9. The camera angle error is within $\pm 1^\circ$ and the viewing distance error is within ± 2 cm.

D. Camera with zooming

Tracking behavior is shown in Fig. 10 where the user adjusts the camera-to-leader desired distance. This is implemented as a feature since stereoscopic cameras normally do not support zoom. The focus point is set to be the end effector position of the leader. It is indicated when a joint in near its maximum and when the collision avoidance force $\boldsymbol{F}_{\text{rep}}$ is nonzero. The desired camera distance is not achieved if the obstacle repulsive force is active. The view angle error is unaffected since the emergency collision control (13) was not needed.

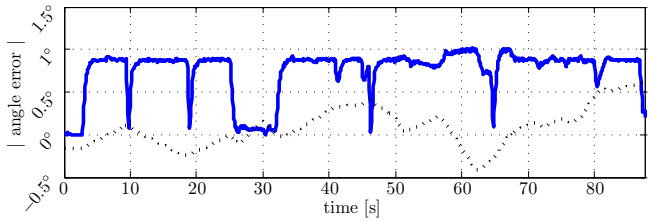
E. Collision avoidance

The camera tracking performance is shown in Fig. 11 as the leader robot tries to crash with the follower robot. The focus point is set to be a fixed point in space not on the leader robot. Regulation toward the fixed point is achieved as long as the emergency collision avoidance routine (13) is inactive. It is indicated in Fig. 11 when the nullspace collision avoidance force is nonzero. The collision avoidance takes priority over the desired view distance as seen in Fig 11b resulting in collision avoidance at the cost of a large view distance error.

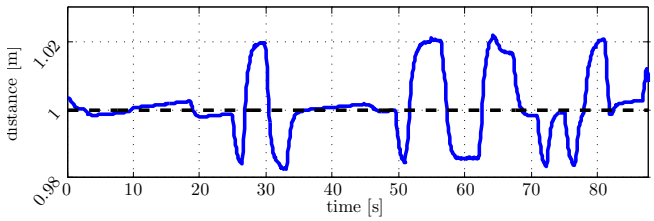
VII. Conclusions and Future Work

A. Conclusions

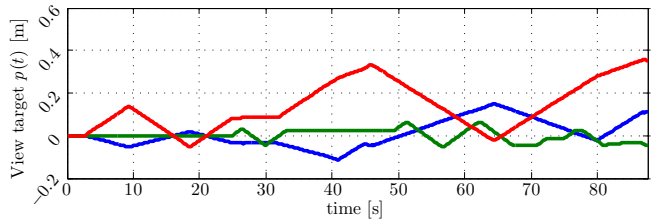
In this paper we have shown how real-time coordinated control of two industrial robot manipulators can be achieved using pseudoinverse redundancy resolution under joint range limitations and collision avoidance constraints. In order to demonstrate the approach, a follower robot equipped with



(a) The absolute angle error $\text{acos}(\boldsymbol{e}_3^T(\boldsymbol{x}-\boldsymbol{p})/\|\boldsymbol{x}-\boldsymbol{p}\|)$ during tracking. A scaled norm of $\boldsymbol{p}$ is shown dotted to indicate the movement of the focus point.

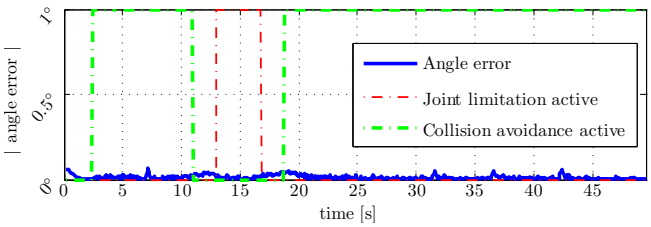


(b) The viewing distance $\|\boldsymbol{x}-\boldsymbol{p}\|$ during tracking.

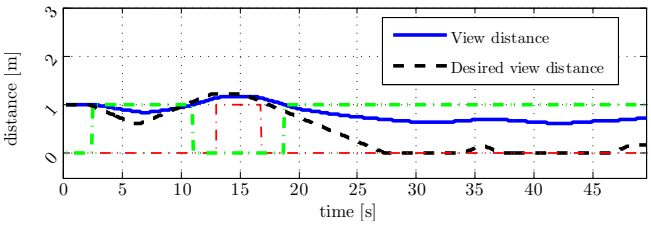


(c) The leader end effector position $\boldsymbol{p}(t)$ initialized at zero during tracking.

Fig. 9: Data from a camera tracking experiment where the initial error is zero.

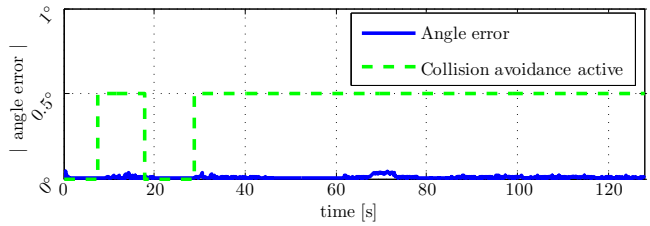


(a) The absolute angle error.

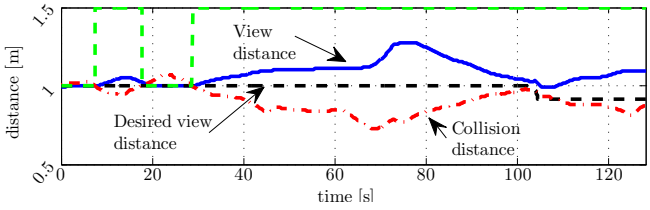


(b) The viewing distance $\|\boldsymbol{x}-\boldsymbol{p}\|$ during tracking for a varying desired distance.

Fig. 10: Data from a camera tracking experiment where the desired viewing distance is varying.



(a) The absolute angle error when the leader is trying to crash with the follower.



(b) The viewing distance $\|\boldsymbol{x} - \boldsymbol{p}\|$ during collision avoidance.

Fig. 11: Data from the collision avoidance experiment.

a stereo vision camera is controlled to follow and point its camera toward a leader robot. Experimental results validated the proposed control methodology. Moreover, a control system architecture is presented which allow for implementation and safe execution of robot controllers in Matlab, together with seamless switching between control of simulated and actual industrial robot manipulators.

The coordination control problem where a follower robot is given the task to monitor a leader robot with a camera was efficiently solved using pseudoinverse redundancy resolution. Joint angle increments were generated for the follower in real-time where only position measurements was available. The coordinated control scheme allows an operator to control the distance between the robot on-line. The proposed control strategy is verified through experiments on two 6-DOF industrial manipulators. The joint increments comply with limitations in joint ranges and collision avoidance constraints. Collision avoidance is first attempted without losing camera focus. If this is not achieved then the camera tracking objective is temporarily relaxed.

The proposed control system architecture employ a System Integrity Module and a custom-built model-based collision checking software in order to ensure safe robot operations. Moreover, the architecture facilitate the use of Matlab for efficient testing of new control algorithms both in simulation and on actual robot manipulators.

In order to develop smaller offshore oil and gas fields, more cost effective solutions are required. Robots may constitute part of such a solution and the results presented in this paper form steps toward such technology progress.

B. Future work

The follower's knowledge of the leader's state is limited to position measurements. Velocity and acceleration estimates of the leader robot may improve the tracking error, especially for high speed camera tracking.

VIII. ACKNOWLEDGMENTS

This work is part of the research project "Next Generation Robotics for Norwegian Industry". The project partners are The Research Council of Norway, SINTEF, NTNU, Statoil, Hydro, Tronrud Engineering, SbSeating, Glen Dimplex Nordic and RobotNorge.

REFERENCES

[1] E. Conkur and R. Buckingham, "Clarifying the definition of redundancy as used in robotics," *Robotica*, vol. 15, no. 5, pp. 583–586, 1997.

[2] A. Liegeois, "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms," *IEEE Trans. Syst., Man, Cybernetics*, vol. 7, pp. 868–871, 1977.

[3] J. Baillieul, J. Hollerbach, and R. Brockett, "Programming and control of kinematically redundant manipulators," in *Proc. IEEE Int. Conf. on Decision and Control*, vol. 23. IEEE, 1984, pp. 768–774.

[4] A. Maciejewski and C. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The International Journal of Robotics Sesearch*, vol. 4, no. 3, pp. 109–117, 1985.

[5] N. Hogan, "Impedance Control: An Approach to Manipulation: Part 1˜ 3," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 107, pp. 1–24, 1985.

[6] D. Whitney, "The mathematics of coordinated control of prosthetic arms and manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 94, p. 303, 1972.

[7] J. Hollerbach and K. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE J. of Robotics and Automation*, vol. 3, no. 4, pp. 308–316, 2002.

[8] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Trans. on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 2002.

[9] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of Intelligent and Robotic Systems*, vol. 3, no. 3, pp. 201–212, 1990.

[10] M. Bjerkeng, K. Pettersen, and E. Kyrkjebø, "Constrained motion control with obstacle avoidance for robot manipulators applied to a telemonitoring system," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Francisco, USA, Sept 2011, submitted.

[11] T. Yoshikawa, "Dynamic hybrid position/force control of robot manipulators. Description of hand constraints and calculation of joint driving force," *IEEE journal of robotics and automation*, vol. 3, no. 5, pp. 386–392, 1987.

[12] M. von Wattenwyl, M. Clerici, and H. Brauchli, "Independent Hybrid Force/Motion Control of Constrained Six-Degrees-of-Freedom Manipulators," *Multibody System Dynamics*, vol. 6, no. 4, pp. 327–342, 2001.

[13] E. Kyrkjebø, P. Liljebäck, and A. A. Transeth, "A robotic concept for remote inspection and maintenance on oil platforms," in *Proc. ASME 28th Int. Conf. on Ocean, Offshore and Arctic Engineering (OMAE 2009)*, Hawaii, USA, May 31 - June 5 2009.

[14] B. Graf and K. Pfeiffer, "Mobile robotics for offshore automation," in *Proc. IARP/EURON Workshop on Robotics for Risky Interventions and Environmental Surveillance*, Benicssim, Spain, 7-8 January 2008.

[15] D. Anisi, J. Gunnar, T. Lillehagen, and C. Skourup, "Robot automation in oil and gas facilities: Indoor and onsite demonstrations," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Oct 2010, pp. 4729–4734.

[16] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. Wiley New Jersey, 2006.

[17] T. Chan and R. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *Robotics and Automation, IEEE Transactions on*, vol. 11, no. 2, pp. 286–292, 2002.

[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, p. 90, 1986.