Modeling, Identification and Control, Vol. 37, No. 1, 2016, pp. 53–62, ISSN 1890–1328

Tracking a Swinging Target with a Robot Manipulator using Visual Sensing

Torstein A. Myhre[1] Olav Egeland[1]

[1]*Department of Production and Quality Engineering, NTNU. NO-7491 Trondheim, Norway. E-mail:* {*torstein.a.myhre,olav.egeland*}*@ntnu.no*

Abstract

In this paper we develop a method for loading parts onto a swinging target using an industrial robot. The orientation of the target is estimated by a particle filter using camera images as measurements. Robust and accurate tracking is achieved by using an accurate dynamic model of the target. The dynamical model is also used to compensate for the time delay between the acquisition of images and the motion response of the robot. The target dynamics is modeled as a spherical pendulum. To ensure robust visual tracking the position of the target mass center is estimated. The method is experimentally validated in a laboratory loading station with a swinging conveyor trolley as target, which is commonly used in industry.

Keywords: Industrial Robots, Particle Filter, Computer Vision for Manufacturing

1 Introduction

Robot vision in industrial applications is typically used where work objects are static or moving at constant velocity, such as when picking from a conveyor belt. A more demanding task is the loading of objects on a swinging conveyor trolley, which is illustrated in Figure 1. In the usual industrial solution objects are loaded on the trolleys manually because they are swinging freely. They are swinging freely to avoid excessive forces and accelerations. This paper presents a method to perform this task automatically, by real-time control of an industrial robot manipulator using an estimate of the trolley orientation computed from camera images with a particle filter.

The controller interfaces of industrial robots are designed to operate at fixed update rates, which can range from 125 Hz (Universal Robots), 250 Hz (KUKA) and up to the kHz level. However, cameras that are used in computer vision applications typically have lower update rates. They are limited by the camera hardware itself and the bandwidth available for image transfer. The images are often transferred using Ethernet or USB, which provides no guarantees of real-time performance by default. These limitations are described in Corke and others (1996) as the “dynamics of visual sensing”. Clearly, for a fixed-rate robot controller to work with the varying frame rates provided by cameras, some method must be employed to compensate for delays and interpolate between images. An approach to compensate for these effects was proposed in Wang et al. (2013), using a dual-rate Kalman filter. A dynamic model was used to predict the target motion at the time instants required for robot control. In Wang et al. (2015) and Lin et al. (2013) it was proposed to identify the parameters of the dynamical model using Expectation-Maximization.

Since the work Isard and Blake (1998) many authors have explored the use of particle filters for visual object tracking. For robotic visual servoing it is specially interesting to look at the work on tracking of rigid bodies in Cartesian space. Particle filter based tracking on the $SE(3)$ group has been investigated using different assumptions on the underlying dynamical model in Kwon et al. (2007), Choi et al. (2011) and Choi and Christensen (2012). In particular, the particle filters were

doi:10.4173/mic.2016.1.5

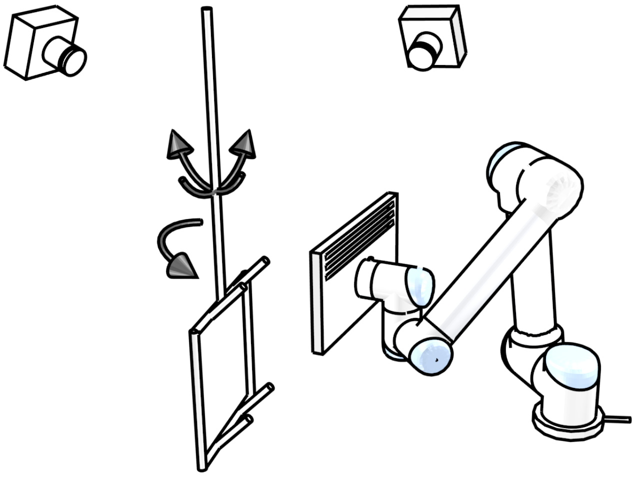


Figure 1: Manual loading of objects onto a swinging conveyor trolley is a common task in industry. The proposed automatic solution is to control the trajectory of the robot manipulator in real-time using an estimate of the trolley motion. An estimate of the trolley motion is found using particle filter based visual tracking.

designed to account for the properties of $SE(3)$. The kinetics were modeled as random walk or with an autoregressive dynamic model. To develop this further, we belive that it may be an advantage to use a kinetic model based on the physical equations of motion. This was done in our previous work Myhre and Egeland (2015) where the dynamics of a spherical pendulum was used. A potential benefit of a physical model is that the assumed noise level in the particle filter can be significantly reduced. One of the challenges in visual tracking is to achieve accuracy despite occlusions and cluttered scenes. A possible solution is to use multiple cameras as in Lippiello et al. (2007) and Kermorgant and Chaumette (2011).

In recent years researchers have demonstrated that particle filter based visual tracking can be used in the robot feedback loop Ibarguren et al. (2014), Chitchian et al. (2013), even though it is considered as a computationally heavy method. The parallel nature of the filter makes it possible to run them on commodity Graphical Processing Units (GPUs) to achieve good performance Choi and Christensen (2013), Concha et al. (2014) and Pauwels et al. (2013).

A dynamical model based on physical principles needs to have accurate parameters in order to be useful. This can be done with parameter estimation. An overview of methods for parameter estimation based on particle filtering can be found in Kantas et al. (2009). The two main on-line approaches are Expectation-Maximization Schön et al. (2011) and gradient ascent Poyiadjis et al. (2011). We used the approach Poyiadjis et al. (2011) in our previous work Myhre and Egeland (2015) to find accurate parameters of a spherical pendulum during visual tracking. This is further developed in the following.

In this paper we propose a method to perform the task illustrated in Figure 1, namely to control a robot manipulator tracking a swinging target using computer vision. The proposed method uses a model based on physical principles with estimated parameters, namely the center of mass position, which enables the robot to accurately track the target even as it accelerates. This is experimentally demonstrated in Section 5. The method proposed in this paper has the same goal as in Lin et al. (2013), which is to compensate for visual sensing dynamics using a model of the motion of the target object. The method presented in Lin et al. (2013) uses a general dynamic model, while we propose to use a physically based dynamical model of a spherical pendulum in order to achieve high performance for the specific application.

In this paper XYZ Euler angles was chosen for the kinematic representation of rotation. The main motivation for this is to simplify and improve the parameter estimation algorithm. The benefits of a coordinate invariant representation with the particle filter was thoroughly discussed in Kwon et al. (2007). In the use case that is presented in this paper, it is unrealistic to consider swinging motions with amplitudes larger than 10°, so the benefits of using a coordinate invariant representation is not as great as in the general case. However, the kinematic convention chosen in this paper has the benefit that the center of mass position naturally can be described using two angular offsets and one linear offset. This enables the parameter estimator to identify the accurate value of the two angular offsets, even when the target is hanging with no velocity (stationary), in which case it is impossible to identify the correct value of the linear offset. In the case of a stationary target, only the two angular offsets are required for visual servoing. A coordinate-free version may be the topic of future research.

This paper presents a novel method for accurate tracking of a swinging target with an industrial robot.

1. The proposed method can handle both stationary and moving targets.

2. A two stage control system is proposed, where one part is running at the rate at which the cameras can deliver images, while the other part is running at the rate required by the robot motion controller. The two stages are connected using a prediction module based on an accurate dynamical model of the swinging target.

3. Experiments are performed to demonstrate that the method can be used for automatic loading of parts onto the swinging target, using a laboratory version of a loading station found in industry.
4. The experiments are performed using standard commercial equipment.

The structure of the paper is as follows: In Section 2 we discuss the preliminaries of particle filtering and parameter estimation, in Section 3 we present a dynamic model and observation model used for tracking, in Section 4 we propose a method for using the state estimate to control an industrial robot manipulator in real-time. Experiments that validate the proposed method are presented in Section 5.

2 Preliminaries

In this paper we consider a non-linear system with additive Gaussian noise

$$\boldsymbol{x}_k = F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) + \boldsymbol{v}_k \tag{1}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ is the state vector at time step k, $\boldsymbol{\theta}$ is a vector of static parameters and $\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ is a noise vector. The probability density of (1) is known as the transition density and can in the case of additive Gaussian noise be written as

$$f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \mathcal{N}(\boldsymbol{x}_k - F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}), \boldsymbol{\Sigma}). \tag{2}$$

An observation is made at each time step k by a camera taking an image, denoted by $\boldsymbol{I}_k$. The relation between $\boldsymbol{I}_k$ and $\boldsymbol{x}_k$ is given by the observation density $g(\boldsymbol{I}_k|\boldsymbol{x}_k)$, which is given in Section 3.2. In this section we first present a particle filter for estimation of the state $\boldsymbol{x}_k$, then a method for estimating the vector of static parameters $\boldsymbol{\theta}$, based on the sequence of camera images.

2.1 Particle Filter

Inferences about the state vector at timestep k can be made using the prediction equation

$$p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k-1}) = \int f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{I}_{1:k-1})\boldsymbol{d}\boldsymbol{x}_{k-1} \tag{3}$$

and the update equation

$$p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k}) = \frac{g(\boldsymbol{I}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k-1})}{\int g(\boldsymbol{I}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k-1})\boldsymbol{d}\boldsymbol{x}_k}. \tag{4}$$

These equations are intractable in general, but particle filters are efficient methods for computing numerical approximations

$$p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^{(i)}), \tag{5}$$

where $\delta(\cdot)$ is the Dirac delta function, $w_k^{(i)}$ are scalar weights and $\boldsymbol{x}_k^{(i)} \in \mathbb{R}^n$.

The specific particle filter used in this paper is known as Sequential Importance Sampling with Resampling which is described in detail in e.g. Cappé et al. (2007) and Doucet and Johansen (2011).

A numerical approximation to the expected value of $p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k})$ can be found as

$$\int \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k})\boldsymbol{d}\boldsymbol{x}_k \approx \sum_{i=1}^{N} w_k^{(i)} \boldsymbol{x}_k^{(i)}. \tag{6}$$

2.2 Estimation of Static Parameters

Methods for sequential estimation of static parameters using particle filters have recently been developed Kantas et al. (2014). An online gradient ascent method is used here,

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{\Gamma}\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{I}_k|\boldsymbol{I}_{1:k-1}), \tag{7}$$

where

$$\boldsymbol{\Gamma} \in \mathbb{R}^{3\times3}. \tag{8}$$

Using the approach presented in Poyiadjis et al. (2011) a set of vectors $\boldsymbol{\alpha}_k^{(i)}$ is found such that

$$\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{I}_k|\boldsymbol{I}_{1:k-1}) \approx \sum_{i=1}^{N} w_k^{(i)}\boldsymbol{\alpha}_k^{(i)} - \sum_{i=1}^{N} w_{k-1}^{(i)}\boldsymbol{\alpha}_{k-1}^{(i)}. \tag{9}$$

The vectors $\boldsymbol{\alpha}_k^{(i)}$ are given by the recursive expression

$$\begin{aligned}\boldsymbol{\alpha}_k^{(i)} = & \frac{\sum_{j=1}^{N} w_{k-1}^{(j)} f(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(j)})}{\sum_{k=1}^{N} w_{k-1}^{(j)} f(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(j)})} \times \Big(\boldsymbol{\alpha}_{k-1}^{(j)} + \\ & \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(j)}) + \nabla_{\boldsymbol{\theta}} \log g(\boldsymbol{I}_k|\boldsymbol{x}_k^{(i)})\Big)\end{aligned} \tag{10}$$

where $\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ and $\nabla_{\boldsymbol{\theta}} \log g(\boldsymbol{I}_k|\boldsymbol{x}_k)$ are gradients of the transition (2) and observation (29) densities respectively, as developed in the next section.

3 Modeling

3.1 Kinematic and Dynamic Modeling

The dynamics of the system is modeled as a spherical pendulum with one additional degree of freedom describing the rotation about the pendulum axis. The configuration can then be described by the XYZ Euler angles ϕ_x, ϕ_y and ϕ_z. When a workpiece is attached to the hanger, the center of mass will be shifted, and the equilibrium position of the hanger will have

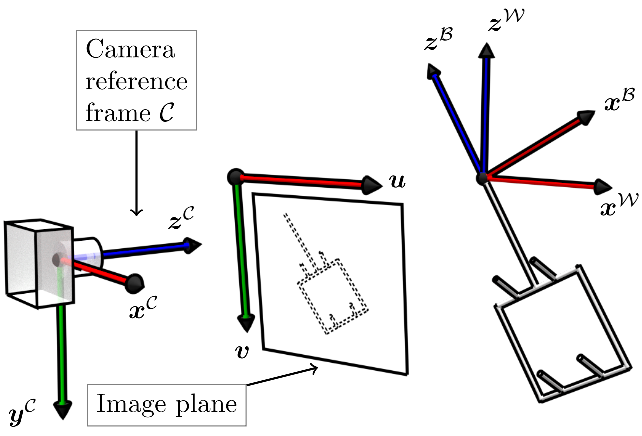

Camera
reference
frame $\mathcal{C}$
$\boldsymbol{z}^{\mathcal{C}}$
$\boldsymbol{x}^{\mathcal{C}}$
$\boldsymbol{y}^{\mathcal{C}}$
Image plane
$\boldsymbol{u}$
$\boldsymbol{v}$
$\boldsymbol{z}^{\mathcal{B}}$
$\boldsymbol{z}^{\mathcal{W}}$
$\boldsymbol{x}^{\mathcal{B}}$
$\boldsymbol{x}^{\mathcal{W}}$

Figure 2: The body reference frame $\mathcal{B}$ is rotating relative to the inertial reference frame $\mathcal{W}$. The geometric trolley model is described by a list of line segments in $\mathbb{R}^3$.

an unknown offset. To account for this uncertainty, we include offset angles θ_1 and θ_2 about the x and y axes, so that the Euler angles become $\Phi_x = \phi_x + \theta_1$, $\Phi_y = \phi_y + \theta_2$ and ϕ_z. Here θ_1 and θ_2 are constant unknown parameters to be identified.

As shown in Figure 2 the world frame is denoted $\mathcal{W}$, and the body-fixed frame is denoted $\mathcal{B}$. The rotation matrix from $\mathcal{W}$ to $\mathcal{B}$ is then given by

$$\boldsymbol{R}_{\mathcal{B}}^{\mathcal{W}} = \boldsymbol{R}_x(\phi_x + \theta_1)\boldsymbol{R}_y(\phi_y + \theta_2)\boldsymbol{R}_z(\phi_z). \tag{11}$$

In the stationary position of the pendulum we have that $\phi_x + \theta_1 = 0$ and $\phi_y + \theta_2 = 0$.

The equations of motion are derived using the Euler-Lagrange equations applied to the Lagrangian

$$\mathcal{L} = \frac{1}{2} m\theta_3^2 \dot{\boldsymbol{r}}_3^{\mathrm{T}} \dot{\boldsymbol{r}}_3 - mg\theta_3 \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \boldsymbol{r}_3 \tag{12}$$

where θ_3 is the unknown constant distance from the pendulum attachment point to the center of mass, and $\boldsymbol{r}_3$ is the last column in $\boldsymbol{R}_{\mathcal{B}}^{\mathcal{W}} = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 \end{bmatrix}$. The distance θ_3 is the third unknown parameter to be identified.

The resulting equations of motion are

$$\begin{aligned} \ddot{\phi}_x &= \frac{2\dot{\phi}_x \dot{\phi}_y \theta_3 \sin 2\Phi_y + 2g \sin \Phi_x \cos \Phi_y}{\theta_3 \cos 2\Phi_y + \theta_3} \\ \ddot{\phi}_y &= \frac{g}{\theta_3} \sin \Phi_y \cos \Phi_x - \frac{1}{2} \dot{\phi}_x^2 \sin 2\Phi_y \\ \ddot{\phi}_z &= 0 \end{aligned} \tag{13}$$

where $g = 9.81\,\mathrm{m\,s^{-2}}$ is the acceleration of gravity.

The state vector of the system (1) is

$$\boldsymbol{x} = \begin{bmatrix} \phi_x & \phi_y & \phi_z & \dot{\phi}_x & \dot{\phi}_y & \dot{\phi}_z \end{bmatrix}^{\mathrm{T}} \tag{14}$$

while the vector of unknown parameters is

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \end{bmatrix}^{\mathrm{T}}. \tag{15}$$

The velocity components in (14) are affected by additive noise, modeled by

$$\boldsymbol{v}_k = \begin{bmatrix} 0 & 0 & 0 & v_k^1 & v_k^2 & v_k^3 \end{bmatrix}^{\mathrm{T}},$$

where the components

$$v_k^1 \sim \mathcal{N}(0, \sigma_1^2), v_k^2 \sim \mathcal{N}(0, \sigma_2^2), v_k^3 \sim \mathcal{N}(0, \sigma_3^2) \tag{16}$$

are samples from Gaussian distributions.

A continuous state space model is found from equations (13), (14) and (15) as

$$\dot{\boldsymbol{x}} = F(\boldsymbol{x}, \boldsymbol{\theta}). \tag{17}$$

The model is discretized in time using the first order Euler method giving $F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})$ in the system (1).

The gradient $\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}_k | \boldsymbol{x}_{k-1})$ is required for parameter estimation in Section 2.2. Let

$$\boldsymbol{\Xi} = \operatorname{diag}(0, 0, 0, 1/\sigma_1^2, 1/\sigma_2^2, 0), \tag{18}$$

then since the transition density is Gaussian,

$$\begin{aligned} &\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}) = \\ &\quad (\boldsymbol{x}_k - F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}))^T \boldsymbol{\Xi} \nabla_{\boldsymbol{\theta}} F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}), \end{aligned} \tag{19}$$

where $\nabla_{\boldsymbol{\theta}} F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})$ is the sensitivity with respect to the parameters $\boldsymbol{\theta}$. The sensitivity is an estimate of the effect variations in the parameter $\boldsymbol{\theta}$ has on $F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})$. From Khalil (2002) the sensitivity is computed by taking derivatives of (17) with respect to the parameters $\boldsymbol{\theta}$ and solving this ode

$$\nabla_{\boldsymbol{\theta}} F_d(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) = \int_{t_{k-1}}^{t_k} \frac{\partial F(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}} + \frac{\partial F(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} dt. \tag{20}$$

3.2 Image Model

It is assumed that the object is a rigid body with orientation given by the rotation matrix $\boldsymbol{R}_{\mathcal{B}}^{\mathcal{W}}$, as illustrated in Fig. 2. A point $\boldsymbol{p}^{\mathcal{W}} \in \mathbb{R}^3$ in frame $\mathcal{W}$ is given in frame $\mathcal{B}$ using

$$\boldsymbol{p}^{\mathcal{W}} = \boldsymbol{R}_{\mathcal{B}}^{\mathcal{W}} \boldsymbol{p}^{\mathcal{B}}. \tag{21}$$

The transformation from the frame $\mathcal{W}$ to $\mathcal{C}$ is given by

$$\boldsymbol{p}^{\mathcal{C}} = \boldsymbol{R}_{\mathcal{W}}^{\mathcal{C}} \boldsymbol{p}^{\mathcal{W}} + \boldsymbol{t}_{\mathcal{W}}^{\mathcal{C}}. \tag{22}$$

The image is a two dimensional array of pixel intensities $I(\boldsymbol{p})$ where $\boldsymbol{p} = \begin{bmatrix} u & v \end{bmatrix}^{\mathrm{T}}$ and u and v are pixel coordinates in the image plane. The camera calibration matrix is

$$\boldsymbol{K} = \begin{bmatrix} k_x & 0 & k_u \\ 0 & k_y & k_v \\ 0 & 0 & 1 \end{bmatrix} \tag{23}$$

where k_x, k_y, k_u and k_v are intrinsic camera calibration parameters. Given a point $\boldsymbol{p}^{\mathcal{C}} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^{\mathrm{T}}$, the coordinates of the point in the image plane is given by

$$p_z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} k_x & 0 & k_u \\ 0 & k_y & k_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{24}$$

or

$$p_z \tilde{\boldsymbol{p}} = \boldsymbol{K}\boldsymbol{p}^{\mathcal{C}} \tag{25}$$

in vector form.

The methods used to find the intrinsic camera parameters k_x, k_y, k_u and k_v and the extrinsic camera parameters $\boldsymbol{R}^{\mathcal{C}}_{\mathcal{W}}$ and $\boldsymbol{t}^{\mathcal{C}}_{\mathcal{W}}$ are described in Section 5.

3.2.1 Trolley Frame Visual Model

The visual model of the trolley is given by N_g line segments. Each line segment j is specified by its endpoints $\boldsymbol{p}^{\mathcal{B}}_j$ and $\boldsymbol{q}^{\mathcal{B}}_j$. For each particle i the rotation matrix $\boldsymbol{R}^{\mathcal{W}}_{\mathcal{B}}$ is computed using the state vector $\boldsymbol{x}^{(i)}_k$. The two points in the image plane corresponding to $\boldsymbol{p}^{\mathcal{B}}_j$ and $\boldsymbol{q}^{\mathcal{B}}_j$ are found in homogeneous pixel coordinates as

$$\tilde{\boldsymbol{p}}_j = \boldsymbol{K}(\boldsymbol{R}^{\mathcal{C}}_{\mathcal{W}}\boldsymbol{R}^{\mathcal{W}}_{\mathcal{B}}\boldsymbol{p}^{\mathcal{B}}_j + \boldsymbol{t}^{\mathcal{W}}_{\mathcal{C}}) \tag{26}$$
$$\tilde{\boldsymbol{q}}_j = \boldsymbol{K}(\boldsymbol{R}^{\mathcal{C}}_{\mathcal{W}}\boldsymbol{R}^{\mathcal{W}}_{\mathcal{B}}\boldsymbol{q}^{\mathcal{B}}_j + \boldsymbol{t}^{\mathcal{W}}_{\mathcal{C}}). \tag{27}$$

The line segment defined by $\boldsymbol{p}^{\mathcal{B}}_j$ and $\boldsymbol{q}^{\mathcal{B}}_j$ is found in pixel coordinates as the line segment from $\tilde{\boldsymbol{p}}_j$ to $\tilde{\boldsymbol{q}}_j$, and can be described as the homogeneous vector $\tilde{\boldsymbol{\ell}}_j = \begin{bmatrix} a & b & c \end{bmatrix}^{\mathrm{T}}$, which is found from the cross product Hartley and Zisserman (2003)

$$\tilde{\boldsymbol{\ell}}_j = \gamma \boldsymbol{S}(\tilde{\boldsymbol{q}}_j)\tilde{\boldsymbol{p}}_j \tag{28}$$

where $\boldsymbol{S}(\tilde{\boldsymbol{q}}_j)$ is the skew symmetric form of $\tilde{\boldsymbol{q}}_j$. A scaling factor γ is used to ensure that the two dimensional vector $\boldsymbol{n}_j = \begin{bmatrix} a & b \end{bmatrix}^{\mathrm{T}}$ is a unit vector. It is noted that in this description the vector $\boldsymbol{n}_j$ is the normal vector to the line segment in image coordinates.

The following observation density is proposed for the particle filter:

$$g(\boldsymbol{I}_k|\boldsymbol{x}_k) = \sum_{i=0}^{N_g} (I(\boldsymbol{p}_j) - I(\boldsymbol{p}_j + \lambda \boldsymbol{n}_j))^2 \tag{29}$$

where $\boldsymbol{p}_j$ is the two-dimensional version of the homogeneous vector $\tilde{\boldsymbol{p}}_j$, $\boldsymbol{n}_j$ is the normal vector to line segment j, and λ is a parameter which was set to $\lambda = 5$, which gives a distance between the points of approximately 5 pixels.

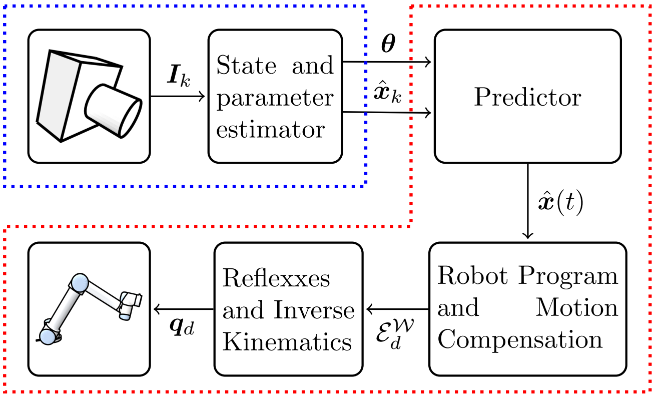

$\boldsymbol{I}_k$
State and parameter estimator
$\boldsymbol{\theta}$
$\hat{\boldsymbol{x}}_k$
Predictor
$\hat{\boldsymbol{x}}(t)$
Robot Program and Motion Compensation
$\mathcal{E}^{\mathcal{W}}_d$
Reflexxes and Inverse Kinematics
$\boldsymbol{q}_d$

Figure 3: The modules comprising the control system. The modules in the blue box are running at the frame rate determined by the cameras. The modules in the red box are running at the fixed rate required by the robot manipulator (125 Hz for UR5).

The parameter vector $\boldsymbol{\theta}$ does not enter into the observation density, which means that

$$\nabla_{\boldsymbol{\theta}} \log g_{\theta}(\boldsymbol{I}_k|\boldsymbol{x}_k) = 0. \tag{30}$$

4 Control System

In this section we describe a control system used for automatic loading of parts on the swinging trolley. The control system structure is visualized in Figure 3. The content of the block containing "State and parameter estimator" was described in Section 2.1 and 2.2. The mean of the state estimate is computed at each time step k

$$\hat{\boldsymbol{x}}_k = \int \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{I}_{1:k})\boldsymbol{d}\boldsymbol{x}_k \tag{31}$$

using (6). The contents of the remaining blocks are described in the following.

4.1 Visual Sensing Dynamics Compensation

The control system illustrated in Figure 3 is logically divided into two parts running at different rates. The modules inside the red dotted polygon are running at the fixed rate required by the robot controller, while the modules inside the blue dotted rectangle are running at the camera rate. As cameras typically come without real-time guarantees, and the frame rate is typically lower than of the robot motion control system, there is a mismatch between the robot controller rate and the camera rate. In order to bridge the gap between the modules running at different rates, the “Predictor” module uses the most recent state-estimate $\hat{\boldsymbol{x}}_k$ and parameter estimate $\boldsymbol{\theta}$, to predict the trolley state $\hat{\boldsymbol{x}}(t)$ at the time instants required to compute set-points for the robot motion controller. The predictor module predicts the trolley state by integrating the system in (17)

$$\hat{\boldsymbol{x}}(t) = \int_{t_k}^{t} F(\boldsymbol{x}, \boldsymbol{\theta}) dt. \tag{32}$$

The result is the predicted transform $\boldsymbol{T}_{\mathcal{B}}^{\mathcal{W}}$ based on the predicted state $\hat{\boldsymbol{x}}(t)$.

4.2 Robot Program and Motion Compensation

In order to move the end-effector smoothly between two reference frames $\boldsymbol{T}_1^{\mathcal{B}} = (\boldsymbol{R}_1^{\mathcal{B}}, \boldsymbol{t}_1^{\mathcal{B}})$ and $\boldsymbol{T}_2^{\mathcal{B}} = (\boldsymbol{R}_2^{\mathcal{B}}, \boldsymbol{t}_2^{\mathcal{B}})$ we define the end-effector reference trajectory

$$\mathcal{E}_d^{\mathcal{B}}(\boldsymbol{T}_1^{\mathcal{B}}, \boldsymbol{T}_2^{\mathcal{B}}, s) = \begin{matrix} \big(\boldsymbol{R}_1^{\mathcal{B}} \exp\left(\log\left(\boldsymbol{R}_{\mathcal{B}}^{1} \boldsymbol{R}_2^{\mathcal{B}}\right)\beta(s)\right), \\ (\boldsymbol{t}_2^{\mathcal{B}} - \boldsymbol{t}_1^{\mathcal{B}})\beta(s) + \boldsymbol{t}_1^{\mathcal{B}}\big) \end{matrix} \tag{33}$$

where $\exp(\cdot)$ is the exponential map $so(3) \rightarrow SO(3)$ and $\log(\cdot)$ is the inverse as defined in Murray et al. (1994). A monotonic function $\beta(s) \in [0, 1]$ is used for interpolation of frames such that excessive acceleration is avoided. We use a linear function of the Logistic function

$$\beta(s) = a \frac{1}{1 + \exp(-ks)} - b. \tag{34}$$

where a and b are choosen so that $\beta(0) = 0$ and $\beta(1) = 1$. The motion of the end-effector is computed using the predicted transform $\boldsymbol{T}_{\mathcal{B}}^{\mathcal{W}}$

$$\mathcal{E}_d^{\mathcal{W}} = \boldsymbol{T}_{\mathcal{B}}^{\mathcal{W}} \mathcal{E}_d^{\mathcal{B}}. \tag{35}$$

4.3 Reflexxes and Inverse Kinematics

The Reflexxes motion libraries Kröger (2011) was used in Cartesian space to filter low amplitude high frequency noise that is part of the particle filter estimate. Set-points for the robot joint controller was thereafter computed using an inverse kinematics procedure.

5 Experiments

5.1 Laboratory set-up

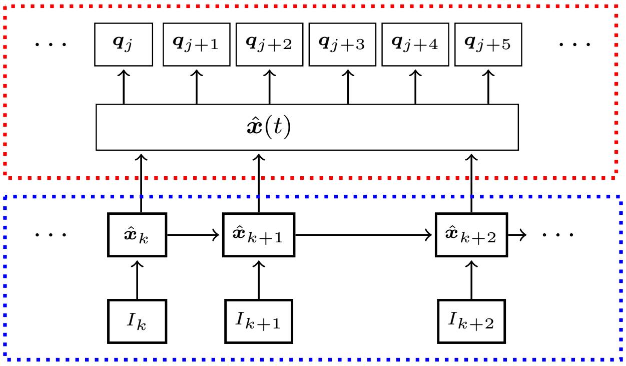


Figure 4: Timing of state estimates $\boldsymbol{x}_k$ in the camera rate (blue box) and the desired joint variables $\boldsymbol{q}_j$ in the robot controller rate (red box).

As shown in Figure 1 a laboratory set-up was built to perform the experiments described in this section. The set-up consisted of two Prosilica GC 1020 Ethernet cameras streaming images to a computer at approximately 35 Hz, which is the fastest they can deliver images at full resolution (1024 × 768 pixels). The computer had a Intel i7-3820 CPU, 16 Gb RAM and a Nvidia Titan graphics card, running Ubuntu Linux 14.10. The Precision Time Protocol was used to achieve synchronization between the clock on the two cameras and the clock on the computer controlling the robot.

A chessboard was mounted on the robot end-effector in order to find the camera calibration parameters required in Section 3.2. For each camera 26 pictures were taken of the chessboard with the robot end-effector in different poses. The intrinsic camera parameters k_x, k_y, k_u and k_v were found using standard camera calibration methods provided by the OpenCV library described in Bradski and Kaehler (2008). The extrinsic camera parameters $\boldsymbol{R}_{\mathcal{W}}^{\mathcal{C}}$ and $\boldsymbol{t}_{\mathcal{W}}^{\mathcal{C}}$ were found using the method described in Park and Martin (1994). The distance from each of the cameras to the trolley frame was approximately 2.2 m.

The differential equations (13) and (20) were discretized using the Euler method. To achieve an accurate estimate of the parameter vector θ it is important that the sensitivity estimate in (20) is accurate. Therefore the step-size for integration was set to 0.0002 s. The two most computationally intensive parts of the particle filter are the observation model and the dynamical model, which therefore were implemented in CUDA in order to run them on the GPU of the Nvidia Titan graphics card.

5.2 Visual Sensing Dynamics Compensation

The experiment in this section was performed to validate the performance of the visual dynamics sensing compensation described in Section 4.1. A 5 s sequence of the state estimate was recorded while the target was swinging, and the resulting ϕ_x component is shown in Figure 5. The graphs show that the output from the

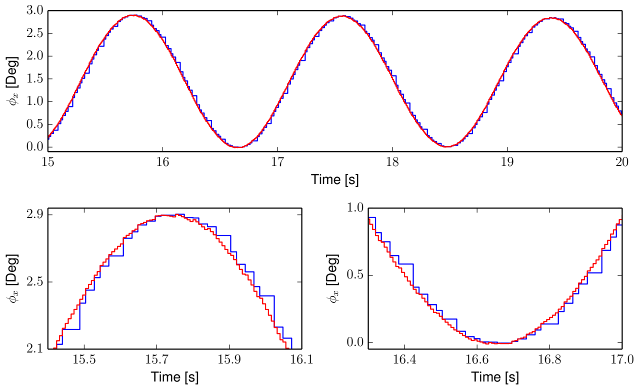


Figure 5: The figure shows the discrete-time state estimate of ϕ_x (in blue), which is updated at the camera rate, and the predicted state (in red), which is in the faster robot rate of 125 Hz.

predictor module provide a smoother and more accurate estimate of the target state, than the estimate coming directly from the state estimator.

5.3 Case Study: Part Loading

In this section we present the experimental validation of the proposed method. We decided to do this by synchronizing the motion of the robot and the trolley, and then using the robot to place a hollow cylinder in loading position. Instead of releasing the grip on the cylinder, the robot then removed the cylinder from the loading position. The idea was then that if the robot could do this without the cylinder coming into contact with the trolley, the synchronization would be accurate within the difference in dimension of the hole in the cylinder and the size of the attachment hook on the trolley. In this case the documented accuracy would be 6 mm.

The target was a trolley hanging from an overhead conveyor, which is used in industrial loading stations commonly operated with manual labour. The experiment was designed to demonstrate that the proposed method also can handle the situations where the position of the mass center changes as objects are attached to the trolley, and that this can be achieved both with moving and stationary targets. To this end, an object was loaded on the trolley so that the center of mass changed to an unknown position, which was estimated by the parameter estimation algorithm. Then the synchronization of robot and trolley motion was demonstrated by letting the robot move the hollow cylinder into loading position and back again without touching the trolley. In this motion the cylinder is very close to the trolley, and a synchronization of 6 mm is validated if the cylinder does not touch the trolley.

The motions that were performed in the experiments are shown in Figure 7. The cylindrical object carried by the robot had inner diameter 22 mm. The trolley frame was welded from steel bars with a square cross section (8 mm × 8 mm). The robot program is described in Figure 6.

Step 1) Estimate the parameter vector θ.

Step 2) Set the time variable $t = 0$ and start tracking using the found parameter vector θ.

Step 3) Move the end-effector from the initial position to the pose $\boldsymbol{T}_1^{\mathcal{B}}$ when $t \in [0, 15)$.

Step 4) Move the end-effector according to $\mathcal{E}_d^{\mathcal{B}}(\boldsymbol{T}_1^{\mathcal{B}}, \boldsymbol{T}_2^{\mathcal{B}}, s(t))$, where

$$s(t) = \begin{cases} (t-15)/5 & \text{if } 15 \leq t < 20 \\ 1 & \text{if } 20 \leq t < 25 \\ \frac{(25-t)}{5} & \text{if } 25 \leq t < 30 \end{cases} \quad (36)$$

Figure 6: Robot program.

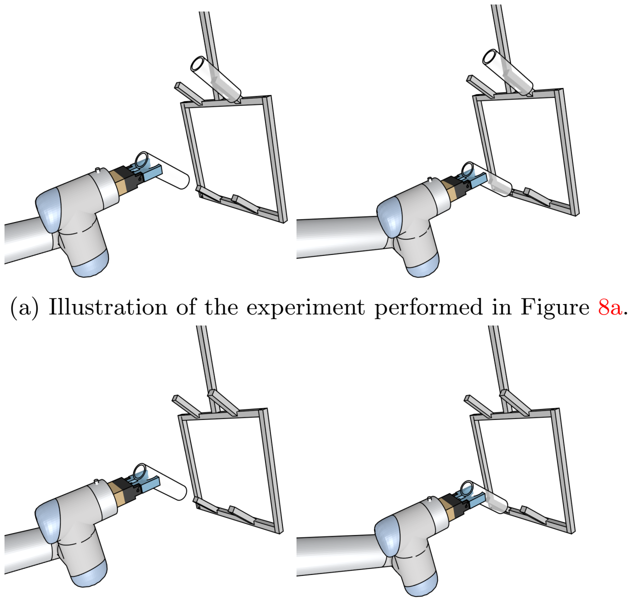
The program was first executed with a stationary target. The results are shown in Figure 8a. In Step 1 the values of the parameters θ_1 and θ_2 converged after 20 s. The parameter θ_3 did not converge because there was no excitation that could be used to identify its value in (7). Estimation was cut off after 25 s. In Step 4 the robot moved the cylindrical object to the loading position on the trolley and back. The states in Figure 8a show that the target was stationary without coming in contact with the robot.

The program was then executed with a moving target. The results are shown in Figure 8b. In Step 1 the values of parameters θ_1, θ_2 and θ_3 converged after 20 s. Estimation was cut off after 25 s. In Step 4 the robot moved the cylindrical object to the loading position on the trolley and back. The states in Figure 8b show that the target motion was smooth without coming in contact with the robot.

5.4 Discussion

The results from the experiment in Section 5.3 shows that

- The mean values of ϕ_x and ϕ_y were $-\theta_1$ and $-\theta_2$, which is consistent with (11).
- The estimated distance to the mass center θ_3 converged when the target was in motion, but not when it was stationary. This was the motivation for choosing the kinematic convention.



(a) Illustration of the experiment performed in Figure 8a.

(b) Illustration of the experiment performed in Figure 8b.

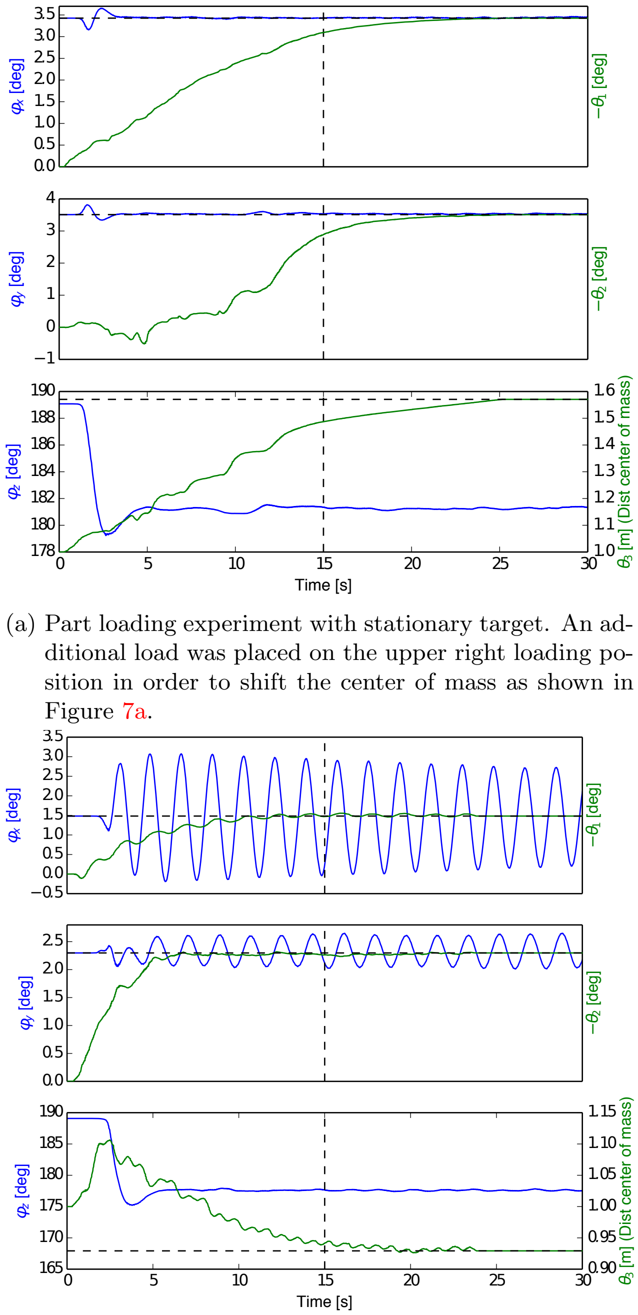
Figure 7: The same experiment is performed twice, first with a heavy load, then without a heavy load. During the experiment the end-effector moves from the pose on the left ($\boldsymbol{T}_1^{\mathcal{B}}$), to the pose on the right ($\boldsymbol{T}_2^{\mathcal{B}}$) and finally back (to $\boldsymbol{T}_1^{\mathcal{B}}$).

- The state vector was not interrupted by external forces during the loading sequence in Step 4, which means that there was no collision between the robot and the target.

- The proposed method for visual servoing worked both with a stationary and a moving target, which is one of the main contributions of this paper.

6 Conclusion

This paper presented a method for accurate tracking of a swinging target using an industrial robot. A dynamical model of the swinging target was used and a method for estimating the parameters describing the mass center position was presented. The model was used to predict the motion of the swinging target, both for compensation of visual sensing dynamics and in the particle filter. The experiments in Section 5 demonstrate that the proposed method can be used to achieve accurate tracking of a swinging target with a robot manipulator. The method was demonstrated on the industrial task of part loading on swinging conveyor trolleys, where the tolerance was less than 6 mm. The method was demonstrated on moving and stationary targets.



(a) Part loading experiment with stationary target. An additional load was placed on the upper right loading position in order to shift the center of mass as shown in Figure 7a.

(b) Part loading experiment with moving target. The additional load was removed as illustrated in Figure 7b.

Figure 8: The graphs illustrates the result of the robot program (Figure 6). The results from Step 1 are shown as the green graphs, which display the evolution of the parameter vector θ. The final values are shown as the horizontal black dashed lines. In Step 4 the end-effector was controlled according to (36), and in the interval (15 s to 30 s) the cylinder was put on the loading position on the trolley. The state vector was recorded and is shown here in blue. It is noted that the states in both Figure 8a and 8b are smooth and show no sign of collision.

Acknowledgment

The research presented in this paper has received funding from the Norwegian Research Council, SFI Offshore Mechatronics, project number 237896.

References

Bradski, G. and Kaehler, A. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Incorporated, 2008.

Cappé, O., Godsill, S. J., and Moulines, E. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 2007. 95(5):899–924. doi:10.1109/jproc.2007.893250.

Chitchian, M., Simonetto, A., van Amesfoort, A. S., and Keviczky, T. Distributed Computation Particle Filters on GPU Architectures for Real-Time Control Applications. *IEEE Trans. Contr. Sys. Techn.*, 2013. 21(6):2224–2238. doi:10.1109/tcst.2012.2234749.

Choi, C. and Christensen, H. RGB-D object tracking: A particle filter approach on GPU. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. pages 1084–1091, 2013. doi:10.1109/iros.2013.6696485.

Choi, C., Christensen, H., and others. Robust 3D visual tracking using particle filtering on the SE (3) group. In *IEEE Int. Conf. on Robotics and Automation*. IEEE, pages 4384–4390, 2011. doi:10.1109/icra.2011.5980245.

Choi, C. and Christensen, H. I. Robust 3D visual tracking using particle filtering on the special Euclidean group: A combined approach of keypoint and edge features. *The International Journal of Robotics Research*, 2012. 31(4):498–519. doi:10.1177/0278364912437213.

Concha, D., Cabido, R., Pantrigo, J., and Montemayor, A. Performance evaluation of a 3D multi-view-based particle filter for visual object tracking using GPUs and multicore CPUs. *Journal of Real-Time Image Processing*, 2014. pages 1–19. doi:10.1007/s11554-014-0483-1.

Corke, P. I. and others. *Visual Control of Robots: high-performance visual servoing*. Research Studies Press Baldock, 1996.

Doucet, A. and Johansen, A. M. A tutorial on particle filtering and smoothing: Fifteen years later. In D. Crisan and B. Rozovskii, editors, *The Oxford handbook of nonlinear filtering*, chapter 24, pages 656–704. Oxford University Press, 2011.

Hartley, R. and Zisserman, A. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge., 2003.

Ibarguren, A., Martínez-Otzeta, J. M., and Maurtua, I. Particle filtering for industrial 6dof visual servoing. *Journal of Intelligent & Robotic Systems*, 2014. 74(3-4):689–696. doi:10.1007/s10846-013-9854-2.

Isard, M. and Blake, A. Condensation—conditional density propagation for visual tracking. *International journal of computer vision*, 1998. 29(1):5–28. doi:10.1023/a:1008078328650.

Kantas, N., Doucet, A., Singh, S. S., and Maciejowski, J. M. An overview of sequential monte carlo methods for parameter estimation in general state-space models. In *Proc. 15th IFAC Symposium on System Identification (SYSID).*, volume 15. pages 774–785, 2009. doi:10.3182/20090706-3-fr-2004.00129.

Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J. M., and Chopin, N. On Particle Methods for Parameter Estimation in State-Space Models. *arXiv:1412.8695, submitted to Statistical Science*, 2014. doi:10.1214/14-sts511.

Kermorgant, O. and Chaumette, F. Multi-sensor data fusion in sensor-based control: Application to multi-camera visual servoing. In *IEEE Int. Conf. on Robotics and Automation*. pages 4518–4523, 2011. doi:10.1109/icra.2011.5979715.

Khalil, H. *Nonlinear systems*, volume 3. Prentice hall, 2002.

Kröger, T. Opening the door to new sensor-based robot applications—The Reflexxes Motion Libraries. In *Proc. IEEE Int. Conf. on Robotics and Automation*. pages 1–4, 2011. doi:10.1109/icra.2011.5980578.

Kwon, J., Choi, M., Park, F. C., and Chun, C. Particle filtering on the Euclidean group: framework and applications. *Robotica*, 2007. 25(06):725–737. doi:10.1017/s0263574707003529.

Lin, C.-Y., Wang, C., and Tomizuka, M. Visual tracking with sensing dynamics compensation using the Expectation-Maximization algorithm. In *Proc. American Control Conference*. IEEE, pages 6281–6286, 2013. doi:10.1109/acc.2013.6580823.

Lippiello, V., Siciliano, B., and Villani, L. Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration. *IEEE Transactions on Robotics*, 2007. 23(1):73–86. doi:10.1109/tro.2006.886832.

Murray, R. M., Li, Z., and Sastry, S. S. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

Myhre, T. A. and Egeland, O. Parameter Estimation for Visual Tracking of a Spherical Pendulum with Particle Filter. In *Proc. IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*. pages 116–121, 2015. doi:10.1109/mfi.2015.7295795.

Park, F. C. and Martin, B. J. Robot sensor calibration: solving AX= XB on the Euclidean group. *IEEE Transactions on Robotics and Automation*, 1994. 10(5):717–721. doi:10.1109/70.326576.

Pauwels, K., Rubio, L., Diaz, J., and Ros, E. Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues. In *IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, pages 2347–2354, 2013. doi:10.1109/cvpr.2013.304.

Poyiadjis, G., Doucet, A., and Singh, S. S. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 2011. 98(1):65–80. doi:10.1093/biomet/asq062.

Schön, T. B., Wills, A., and Ninness, B. System identification of nonlinear state-space models. *Automatica*, 2011. 47(1):39–49. doi:10.1016/j.automatica.2010.10.013.

Wang, C., Lin, C.-Y., and Tomizuka, M. Visual servoing for robot manipulators considering sensing and dynamics limitations. In *Proc. ASME Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, pages 21–23, 2013. doi:10.1115/dscc2013-3833.

Wang, C., Lin, C.-Y., and Tomizuka, M. Statistical learning algorithms to compensate slow visual feedback for industrial robots. *Journal of Dynamic Systems, Measurement, and Control*, 2015. 137(3). doi:10.1115/1.4027853.