REAL-TIME TIME-OPTIMAL CONTROL FOR A NONLINEAR CONTAINER CRANE USING A NEURAL NETWORK

T.J.J. van den Boom, J.B. Klaassens and R. Meiland
Delft Center for Systems and Control
Mekelweg 2, 2628 CD Delft, The Netherlands

Keywords: Time-optimal crane control, Nonlinear Model Predictive Control, Optimization, binary search algorithm, neural networks, Bayesian regularization.

Abstract: This paper considers time-optimal control for a container crane based on a Model Predictive Control approach. The model we use is nonlinear and it is planar, i.e. we only consider the swing (not the skew) and we take constraints on the input signal into consideration. Since the time required for the optimization makes time-optimal not suitable for fast systems and/or complex systems, such as the crane system we consider, we propose an off-line computation of the control law by using a neural network. After the neural network has been trained off-line, it can then be used in an on-line mode as a feedback control strategy.

1 INTRODUCTION

The need for fast transport of containers from quay to ship and from ship to quay, is growing more and more. Since ships and their container capacity grow larger, a more time efficient manner of (un)loading containers is required. Shipping companies focus on maximizing the sailing hours and reducing the hours spent in port. A longer stay in port will eliminate the profit gained at sea for the large vessels and can hardly be considered as an option.

Much research has been done on the crane modelling and control (Marttinen et al., 1990),(Fliess et al., 1991),(Hämäläinen et al., 1995), (Bartolini et al., 2002), (Giua et al., 1999) however most models are linearized. In this paper we study time-optimal control for a container crane using a nonlinear model. The drawback of time-optimal control, in the presence of constraints, is its demand with respect to computational complexity. This doesn't make time-optimal control suitable for fast systems, such as the crane system. To overcome this problem a neural network can be used. It can be trained off-line to 'learn' the control law obtained by the time-optimal controller. After the neural network has been trained off-line, it can then be used in an on-line mode as a feedback control strategy. In Nonlinear Model Predictive Control (MPC) an off-line computation of the control law using a feed-forward neural network was already proposed by (Parisini and Zoppoli, 1995). The off-line approach was also followed in (Pottman and Seborg, 1997), where a radial basis function network was used to 'memorize' the control actions. In this paper we extend these ideas to time-optimal control.

Section 2 describes the continuous-time model of the crane and the conversion from the continuous-time model to a discrete-time model. Section 3 discusses time-optimal control. Section 4 gives an outline of a feedforward network and discuss the best architecture of the neural network with respect to the provided training data. Section 5 gives conclusions about how well the time-optimal controller performs in real time.

2 CRANE MODEL

A dynamical crane model is presented in this section. A schematic picture of the container crane is shown in figure 1. The container is picked up by a spreader, which is connected to a trolley by the hoisting cables. One drive in controlling the motion of the trolley and another drive is controlling the hoisting mechanism. The electrical machines produce the force F_T acting on the trolley and the hoisting force F_H and providing the motion of the load. The dynamics of the electrical motors is not included in the model. The combina-

J. J. van den Boom T., B. Klaassens J. and Meiland R. (2005).
REAL-TIME TIME-OPTIMAL CONTROL FOR A NONLINEAR CONTAINER CRANE USING A NEURAL NETWORK.
In *Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics*, pages 39-44
DOI: 10.5220/0001179800390044

tion of the trolley and the container is identified as a two-sided pendulum. The elastic deformation in the cables and the crane construction is neglected. The load (spreader and container) is presented as a mass m_c hanging on a massless rope. Friction in the system is neglected. Only the swing of the container is considered while other motions like skew are not taken into account. Further, the sensors are supposed to be ideal without noise. The influence of wind and sensor noise can be included in the model as disturbances.

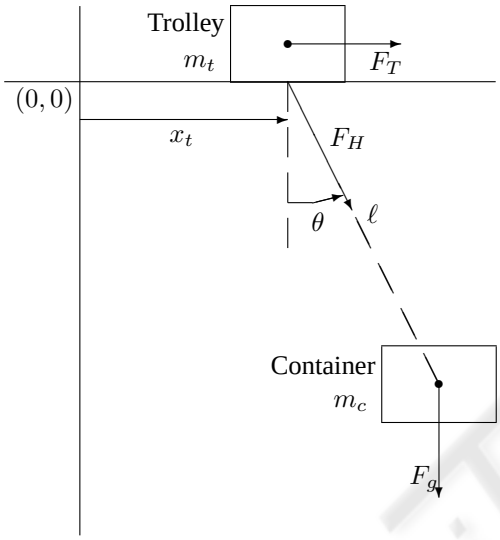

Figure 1: Jumbo Container Crane model

The continuous time model is presented by the following equations:

$$\ddot{x}_t = \frac{m_c G_h g \sin\theta\cos\theta + m_c G_h l\dot{\theta}^2 \sin\theta}{(m_c+G_h)(m_t+G_t)+G_h m_c(1-\cos^2\theta)} + \frac{F_T(m_c+G_h) - m_c F_H \sin\theta}{(m_c+G_h)(m_t+G_t)+G_h m_c(1-\cos^2\theta)} \quad (1)$$

$$\ddot{\theta} = \frac{-\ddot{x}_t\cos\theta - 2\dot{l}\dot{\theta} - g\sin\theta}{l} \quad (2)$$

$$\ddot{l} = \frac{F_H - m_c\ddot{x}_t\sin\theta + m_c l\dot{\theta}^2 + m_c g\cos\theta}{m_c+G_h} \quad (3)$$

where x_t is position of the trolley, θ is the swing angle, l is the length of the cable, m_c is the container mass, m_t is the trolley mass, G_t is the virtual trolley motor mass, G_h is the virtual hoisting motor mass, F_T transversal force and F_H is the hoisting force.

By defining the following state and control signals

$$z = \begin{bmatrix} x_t \\ \dot{x_t} \\ \theta \\ \dot{\theta} \\ l \\ \dot{l} \end{bmatrix}, \quad u = \begin{bmatrix} F_T \\ (F_H - F_{H0}) \end{bmatrix},$$

where $F_{H0} = -m_c g$ is used to compensate gravity of the container, we obtain the continuous dynamic system in the following form:

$$\dot{z}(t) = f(z(t), u(t)) \quad (4)$$

Discrete-time model

Since the controller we will use is discrete, a discrete-time model is needed. We have chosen Euler's method because it is a fast method. The Euler approximation is given by:

$$z(k+1) = z(k) + T \cdot f(k, z(k), u(k)) \quad (5)$$

where the integration interval $\triangle t$ is the sampling time T.

3 TIME-OPTIMAL CONTROL

In this paper we consider time-optimal control for the crane. Some papers recommend the planning of a time-optimal trajectory and use this as a reference path for the container to follow ((Gao and Chen, 1997), (Kiss et al., 2000), (Klaassens et al., 1999)). We have chosen not to determine a pre-calculated time-optimal path and subsequently use this as a reference, in stead we calculate the time-optimal path using a two step iteration. In a first step we propose a specific time interval N and evaluate if there is a feasible solution that brings the container to the desired end-position $(x_{c,\mathrm{des}}, y_{c,\mathrm{des}})$ within the proposed time interval N. In the second step we enlarge the time interval if no feasible solution exists, or we shrink the interval if there is a feasible solution. We iterate over step 1 and step 2 until we find the smallest time interval N_{opt} for which there exists a feasible solution.

First step:
To decide, within the first step, whether there is a feasible solution for a given time interval N, we minimize a Model Predictive Control (MPC) type cost-criterion $J(u,k)$ with respect to the time interval constraint and additional constraint for a smooth operation of the crane. In MPC we consider the future evolution of the system over a given prediction period $[k+1, k+N_p]$, which is characterized by the prediction horizon N_p (which is much larger than the proposed time interval), and where k is the current sample step. For the system (5) we can make an estimate

$\hat{z}(k+j)$ of the output at sample step $k+j$ based on the state $z(k)$ at step k and the future input sequence $u(k), u(k+1), \ldots, u(k+j-1)$. Using successive substitution, we obtain an expression of the form

$$\hat{z}(k+j) = F_j(z(k), u(k), u(k+1), \ldots, u(k+j-1))$$

for $j = 1, \ldots, N_p$. If we define the vectors

$$\tilde{u}(k) = \left[u^{\mathrm{T}}(k) \quad \ldots \quad u^{\mathrm{T}}(k+N_p-1) \right]^{\mathrm{T}} \tag{6}$$

$$\tilde{z}(k) = \left[\hat{z}(k+1) \quad \ldots \quad \hat{z}(k+N_p) \right]^{\mathrm{T}} , \tag{7}$$

we obtain the following prediction equation:

$$\tilde{z}(k) = \tilde{F}(z(k), \tilde{u}(k)) \ . \tag{8}$$

The cost criterion $J(u,k)$ used in MPC reflects the reference tracking error ($J_{\text{out}}(\tilde{u},k)$) and the control effort ($J_{\text{in}}(\tilde{u},k)$):

$$\begin{aligned} J(\tilde{u},k) &= J_{\text{out}}(\tilde{u},k)(k) + \lambda J_{\text{in}}(\tilde{u},k)(k) \\ &= \sum_{j=1}^{N_p} |\hat{x}_c(k+j) - x_{c,\text{des}}|^2 + |\hat{y}_c(k+j) - y_{c,\text{des}}|^2 \\ &\quad + \lambda |u(k+j-1)|^2 \end{aligned} \tag{9}$$

where $x_c = \hat{z}_1 + \hat{z}_5 \sin \hat{z}_3$ is the x-position of the container, $y_c = \hat{z}_5 \cos \hat{z}_3$ is the y-position of the container, and λ is a nonnegative integer. From the above it is clear that $J(k)$ is a function of $\tilde{z}(k)$ and $\tilde{u}(k)$, and so is a function of $z(k)$ and $\tilde{u}(k)$.

In practical situations, there will be constraints on the input forces applied to the crane:

$$\begin{array}{rcl} -F_{T\,\max} \leq & u_1 & \leq F_{T\,\max}, \\ F_{H\,\max} - F_{H0} \leq & u_2 & \leq -F_{H0}. \end{array} \tag{10}$$

where, because of the sign of F_H, we have $F_{H\,\max} < 0$ and $F_{H0} = -m_c g < 0$. Further we have the time interval constraints that

$$\begin{array}{rcl} |\hat{x}_c(N+i) - x_{c,\text{des}}| & \leq & \epsilon_x, \ \ i \geq 0 \\ |\hat{y}_c(N+i) - y_{c,\text{des}}| & \leq & \epsilon_y, \ \ i \geq 0 \end{array} \tag{11}$$

which means that at the end of the time interval N the container must be at its destination with a desired precision ϵ_x and ϵ_y, respectively.

Consider the constrained optimization problem to find at time step k a $\tilde{u}(k)$ where:

$$\tilde{u}^*(k) = \arg\min_{\tilde{u}} J(\tilde{u},k)$$

subject to (10) and (11). Note that the above optimization is a nonlinear optimization with constraints. To reduce calculation time for the optimization we can rewrite the constrained optimization problem into an unconstrained optimization problem by introducing auxiliary input variables for the force constraints and penalty functions to account for the time interval constraint. For the force constraints we consider the auxiliary inputs v_1 and v_2:

$$\begin{aligned} u_1 &= \alpha \arctan\left(\frac{v_1}{\alpha}\right) \\ u_2 &= \begin{cases} \beta \arctan\left(\frac{v_2}{\beta}\right), & v_2 < 0 \\ v_2, & 0 < v_2 < (\beta\pi/2) \\ \gamma + \beta \arctan\left(\frac{v_2-\gamma}{\beta}\right), & v_2 > (\beta\pi/2) \end{cases} \end{aligned}$$

where $\alpha = 2F_{T\,\max}/\pi$, $\beta = 2(F_{H\,\max} - F_{H0})/\pi$ and $\gamma = F_{H\,\max} - 2F_{H0}$. Note that for all $v_1, v_2 \in \mathbb{R}$ input force constraints (10) will be satisfied.

For the time interval constraints we define the penalty function:

$$\begin{aligned} J_{\text{pen}}(\tilde{u},k) &= \sum_{j=N-k}^{N_p} \mu |\hat{x}_c(k+j) - x_{c,\text{des}}|^2 \\ &\quad + \mu |\hat{y}_c(k+j) - y_{c,\text{des}}|^2 \end{aligned} \tag{12}$$

where $\mu \gg 1$. Beyond the time interval (so for $k+j \geq N$) the influence of any deviation from the desired end point is large and the container position and speed must then be very accurate.

Instead of the constrained optimization problem we have now recast the problem as an unconstrained optimization problem at time step k:

$$\tilde{v}^*(k) = \arg\min_{\tilde{v}} J(\tilde{u}(\tilde{v}),k) + J_{\text{pen}}(\tilde{u}(\tilde{v}),k)$$

where

$$\tilde{v}(k) = \left[v^{\mathrm{T}}(k) \quad \ldots \quad v^{\mathrm{T}}(k+N_p-1) \right]^{\mathrm{T}}$$

For the optimization we use an iterative optimization algorithm where in each iteration step we first select a search direction and then we perform a line search, i.e., an optimization along the search direction. The search direction is according the Broyden-Fletcher-Goldfarb-Shanno method and for the line search we have chosen a mixed quadratic and cubic polynomial method.

Second step:
In the first step we have a constant penalty function shifting point N, which has to be chosen differently for every different initial state and steady state. When we have chosen a value for N for a certain initial state and steady state such that the states converge, we can lower the value of N. On the other hand, when we have chosen a value for N for a certain initial state and steady state such that the states do not converge within the allowed region, we have to increase the value of N. When we have found the optimal value $N = N_{\text{opt}}$ if for N there exists a feasible solution, and reduction of N will lead to an infeasible problem. In other words, The determination of N_{opt} has

become a feasibility study. To determine the optimal value N_{opt} for each different initial state z_0 and steady state $(x_{c,\text{des}}, y_{c,\text{des}})$ in an efficient way, we have implemented a simple binary search algorithm.

4 NEURAL NETWORK

Since the time required for the optimization makes time-optimal control not suitable for fast systems, we propose an off-line computation of the control law using a neural network. We assume the existence of a function that maps the state to the optimal control action, and this function is continuous. Continuous functions can be approximated to any degree of accuracy on a given compact set by feedforward neural networks based on sigmoidal functions, provided that the number of neural units is sufficiently large. However, this assumption is only valid if the solution to the optimization problem is unique. After the neural network controller has been constructed off-line, it can then be used in an on-line mode as a feedback control strategy. Because the network will always be an approximation, it cannot be guaranteed that constraints are not violated. However, input constraints, which are the only constraints we consider, can always be satisfied by limiting the output of the network.

Training of the neural network

We have covered the workspace of the crane as can be seen in Table 1. We have considered all initial

Table 1: Values of x_0 and x_{des}

x_{t_0}	=	0	[m]
$x_{t,\text{des}}$	=	$\{0, 5, 10, \ldots, 60\}$	[m]
l_0	=	$\{5, 10, 15, \ldots, 50\}$	[m]
l_{des}	=	$\{5, 10, 15, \ldots, 50\}$	[m]

speeds zero, i.e. $\dot{x}_{t_0}, \dot{\theta}_0, \dot{l}_0$ as well as the swing angle θ_0 are zero. The initial state for the trolley, x_{t_0}, is always zero, and the steady state is within the range $0 \leq x_{t,\text{des}} \leq 60$ m , with steps of 5 m. The dynamical behavior of the crane depends on the distance of the trolley travelling $x_t - x_{t,\text{des}}$ and not on its position. This explains why we only consider $x_{t_0} = 0$ m.

We don't consider simulations where we start and end in the same states, or in other words, where we stay in equilibrium. Thus the total amount of different combinations of initial states x_0 and steady states x_{des} is $13 \times 10 \times 10 - 10 = 1290$.

It is of utmost importance to keep the number of inputs and outputs of the neural network as low as possible. This to avoid unnecessary complexity with respect to the architecture of the neural network. Notice that most of the steady states we use for the control problem, are always zero and can be disregarded for the input signal of the neural network. The only exceptions are $x_{t,\text{des}}$ and l_{des}. Furthermore, we can reduce the number of inputs by taking the distance of the trolley travelling $x_t - x_{t,\text{des}}$, while still providing the same dynamic behavior. We cannot reduce the number of the outputs, hence for the minimum number of inputs (z) and outputs (y) we have:

$$z = \begin{bmatrix} x_t - x_{t,\text{des}} \\ \dot{x}_t \\ \theta \\ \dot{\theta} \\ l \\ \dot{l} \\ l_{\text{des}} \end{bmatrix}, \; u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

We can reduce the dimension of the input vector even more by using principal component analysis as a preprocessing strategy. We eliminate those principal components which contribute less than 2 percent. The result is that the total number of inputs now is 6 in stead of 7.

We have trained the neural network off-line with the Levenberg-Marquardt algorithm. We have used one hidden layer and we have used Bayesian regularization to determine the optimal setting of hidden neurons. For more detail about Bayesian regularization we refer to (Mackay, 1992) and (Foresee and Hagan, 1997).

Table 2: Bayesian regularization results for a 6-m_1-2 feedforward network

m_1	$\mathcal{E}_{Tr}$	$\mathcal{E}_{Tst}$	$\mathcal{E}_{Val}$	$\mathcal{E}_w$	W_{eff}
5	42318	26759	14182	176	46.9
10	34463	29379	11568	226	90.6
20	24796	32502	8425	2164	180
30	24318	32819	8270	1219	268
40	21636	33573	7411	1099	357
50	18726	34617	6420	2270	445
60	19830	34152	6831	813	535
70	3462	7315	1424	1453	618
80	3599	7350	1473	828	704
90	3337	7459	1409	1232	793
100	3404	7473	1459	923	875
110	3225	7371	1401	1100	964
120	3237	7401	1437	1005	1046
130	3512	7281	1415	982	977

For the results we refer to Table 2 where m_1 denotes the number of neurons of the first (and only) hidden layer, $\mathcal{E}_{Tr}$, $\mathcal{E}_{Tst}$ and $\mathcal{E}_{Val}$ denote the sum of squared errors on the training subset, test subset and on the validation subset respectively. The sum of

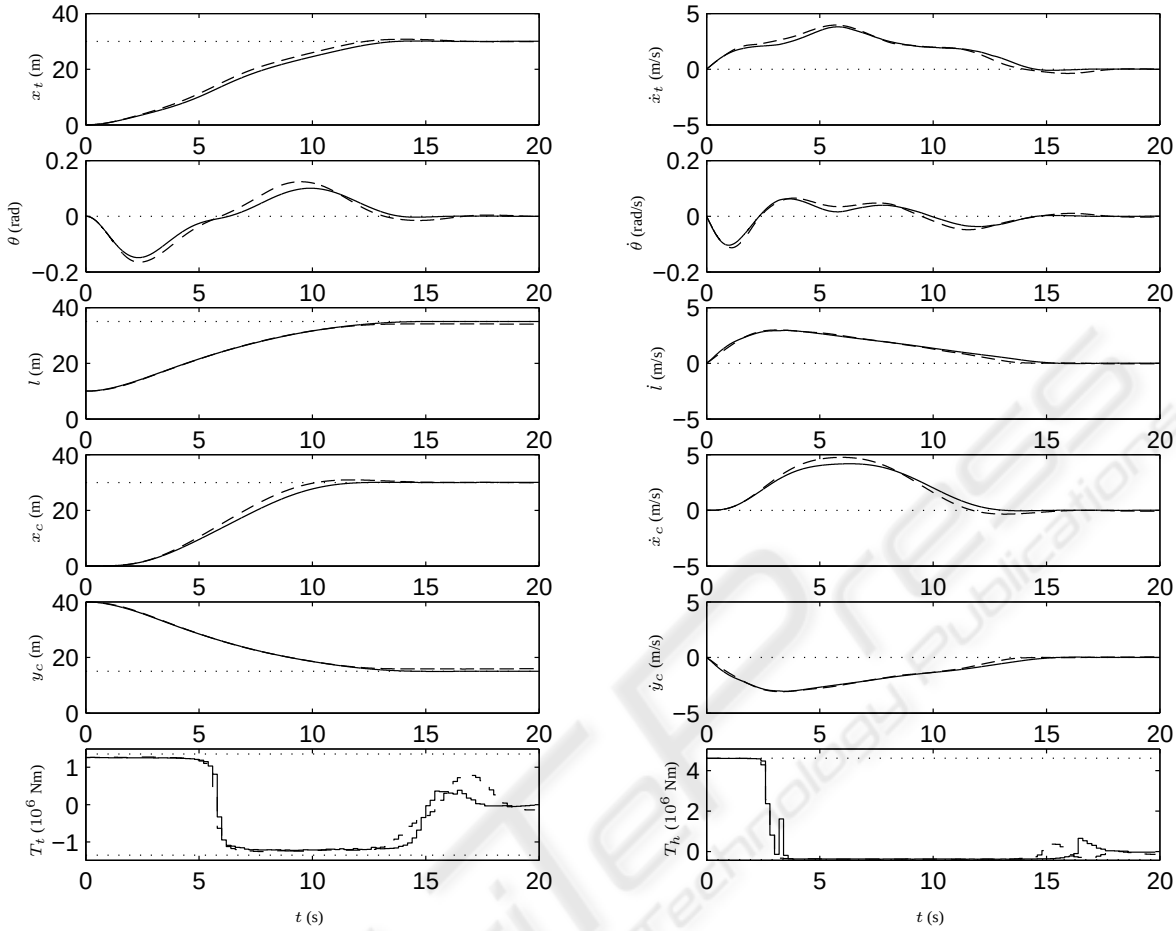

Figure 2: Comparison of simulation results between the time-optimal controller (solid line) and the neural network approximation (dashed line)

squares on the error weights is denotes by $\mathcal{E}_w$ and W_{eff} is the effective number of parameters.

We have tested the neural network controller as a feedback control strategy in an online mode. Figure 2 shows a comparison between the neural network controller and the time-optimal controller where the dashed line denotes the simulation for the time-optimal controller and the solid line denotes the neural network simulation. The result seems satisfactory. The total cpu time of the neural network was 2.5 s and the total cpu time of the time-optimal controller was 17 minutes and 58 seconds. The neural network controller can easily be implemented in an online mode as a feedback control strategy.

5 DISCUSSION

In this paper an implementation of a time-optimal controller for a planar crane system is presented, based on an MPC approach. A nonlinear state space system was used for a model and we have implemented on the inputs. We have trained a neural network off-line with the training data obtained from the time-optimal controller. We have used Bayesian regularization to determine the optimal settings of the total number of hidden neurons. The trained neural network can be used in an online feedback control strategy.

In future research we will search methods to obtain the data, necessary for training the neural networks, in an efficient way, and to avoid redundancy. Further we will include the skew motion of the container, and introduce trajectory constraints to prevent collision of the container with other objects.

REFERENCES

Bartolini, G., Pisano, A., and Usai, E. (2002). Second-order sliding-mode control of container cranes. *Automatica*, 38:1783–1790.

Fliess, M., Lévine, J., and Rouchon, P. (1991). A simplified approach of crane control via a generalized state-space model. Proceedings of the 30th Conference on Decision and Control, Brighton, England.

Foresee, F. and Hagan, M. (1997). Gauss-newton approximation to bayesian learning. *Proceedings of the 1997 International Joint Conference on Neural Networks*, pages 1930–1935.

Gao, J. and Chen, D. (1997). Learning control of an overhead crane for obstacle avoidance. Proceedings of the American Control Conference, Albuquerque, New Mexico.

Giua, A., Seatzu, C., and Usai, G. (1999). Oberserver-controller design for cranes via lyapunov equivalence. *Automatica*, 35:669–678.

Hämäläinen, J., Marttinen, A., Baharova, L., and Virkkunen, J. (1995). Optimal path planning for a trolley crane: fast and smooth transfer of load. *IEEE Proc.-Control Theory Appl.*, 142(1):51–57.

Kiss, B., Lévine, J., and Mullhaupt, P. (2000). Control of a reduced size model of us navy crane using only motor position sensors. In: *Nonlinear Control in the Year 2000*, Editor: Isidori, A., F. Lamnabhi-Lagarrigue and W. Respondek. Springer, New York, 2000, Vol.2. pp. 1-12.

Klaassens, J., Honderd, G., Azzouzi, A. E., Cheok, K. C., and Smid, G. (1999). 3d modeling visualization for studying control of the jumbo container crane. *Proceedings of the American Control Conference, San Diego, California*, pages 1754–1758.

Mackay, D. (1992). Bayesian interpolation. *Neural Computation*, 4(3):415–447.

Marttinen, A., Virkkunen, J., and Salminen, R. (1990). Control study with a pilot crane. *IEEE Transactions on education*, 33(3):298–305.

Parisini, T. and Zoppoli, R. (1995). A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica*, 31(10):1443–1451.

Pottman, M. and Seborg, D. (1997). A nonlinear predictive control strategy based on radial basis function models. *Comp. Chem. Eng.*, 21(9):965–980.