Control of Depth of Anaesthesia using locally weighted learning methods

André Falcão
Instituto Superior Técnico
Lisboa, Portugal

***Abstract*—This paper addresses the control of depth of anaesthesia (DoA) with a novel approach. For control proposes DoA is modeled as a non-linear dynamic system. The approach consists in approximating the non-linear system by a set of locally linear systems obtained from one of two locally weighted learning (LWL) methods: RFWR (Receptive fields weighted regression) [1] or its most recent development LWPR (Locally weighted projection regression) [2] [3]. The estimated local models are then used to synthesize local controllers that trough a controller fusion strategy, provide control to the the non-linear system.**

***Index Terms*—Control, Adaptation, Learning, Anaesthesia, Biomedical Engineering, Multiple Models.**

I. Introduction

The work presented in this paper was developed in the scope of project IDeA (http://ramses.inesc.pt/IDEA/), a multidisciplinary research project with the objective of developing an autonomous system to administer anaesthesia in a patient subject to surgery.

The introduction of intravenous anesthetic drugs in the mid 20th century introduced major changes in anaesthesia. Since then, a number of increasingly effective drugs have been introduced that act within minutes and able to block specific human body mechanisms such as cognition, awareness, memory, stress response or muscle movement [4]. This fact allied to the computer revolution beginning in the 1980´s that brought actuators, such as computer controlled syringes, sensors such as electroencephalogram (EEG), and computer control theory, made anaesthesia automation an active research topic.

Three main motivations arise when talking about anaesthesia automation. The first is precision: an automatic control scheme may have, a considerably faster response time than manual control and provide a better regulation behavior. Precision is essential because it allows less drug underdosage or overdosage problems which are related to a higher probability of patient complications during or after surgery.

Allied to precision comes the economical motivation since precise regulation means less unnecessary drug administration. In terms of drug cost this is not an important issue. However anaesthesia automation has potentially a major economic impact in the resolution of the periods in which surgery rooms are needed for a given patient.

The third motivation and possibly the most important is to provide an anesthetist with an "autopilot" for the repetitive and time consuming task of continuous drug administration, thereby allowing the practitioner to concentrate on other high level tasks and reducing the probability of human error.

Genereal anaesthesia can be divided into three main components each related to a specific physical effect:

1) **Areflexia** – Reflex movement must be avoided. Usually, drugs like *attracurium*, are administered to induce and maintain a certain level of paralysis.
2) **Unconsciousness** – In many cases to avoid stress the patient must not be aware of what is happening during surgery. A certain level of unconsciousness is achieved and maintained trough the administration of drugs like *propofol*. The drug is administered intravenously, being adequate to automatic control.
3) **Analgesia** – the patient should not feel any *noxious stimuli*, commonly known as "pain". To suppress it, analgesic dugs like *remifentanil*, are administered.

To achieve a fully automated system all three components must be controlled. However all of the work and advances done so far are concentrated on the areflexia and unconsciousness components because they can both be measured trough electromyogram (EMG) and Bi-Spectral index (BIS) respectively (other sensor are available as well). In opposition "pain" level measurement involves cross referencing data coming from multiple sensors, such as, EEG or cardiovascular functions and also some amount of empirical evaluation from the practitioner based on work experience.

Even for the measurable components, devising an efficient and robust control strategy for such a complex biological system as the human body constitutes a hard task. Two main factors rise problems:

- Several mathematical patient models were derived in extensive previous work [5] [6] [7] but are, as expected, non-linear.
- High patient to patient variability exists leading to high parametric uncertainty in the patient models.

For the unconsciousness component (DoA) a third problem adds an additional difficulty. The drug used to suppress pain, *remifentanil*, interacts with *propofol* changing its effect.

Another aspect worth mentioning is related to the patient model structure itself. As described in [7] compartmental models are normally used as they possess a simpler structure and thus less parameters to be determined. The downside to the previously mentioned choice is a loss in model accuracy, that could degrade performance in control strategies relying

on such a structure.
This work will study a different approach to the DoA control problem using the already mentioned LWL algorithms to learn the patients dynamic behavior: RFWR [1] or LWPR [2] [3].
Using LWL methods for DoA presents two main advantages:

- It provides the possibility of online adaptation, which is essential to tackle the problem of variability from patient to patient and parameter variation due to drug interaction.
- No model structure is assumed. LWL methods constitute a non parametric approach.

The paper is organized as follows: after the introduction, the patient dynamics for DoA is described in section II, in section III the working principle for the two LWL methods is stated, section IV describes the use of the LWL methods for system identification, based on the previously mentioned identification the control strategy is defined on section V, followed by simulation results and conclusions on chapter VI.

II. The DoA model

The objective for this section is to provide further insight on the patient dynamics for DoA and to define the particular DoA model used to perform the control simulations in this work.

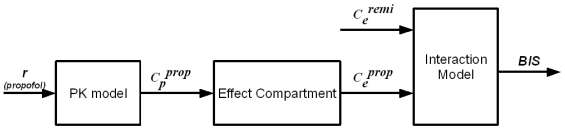

Figure 1. General structure for the DoA model

The DoA model [5] is divided into three main parts (figure 1). The Pharmaco-Kinetic model describes how the hypnotic drug (*propofol*) administered with an infusion rate r, spreads through the blood and tissue resulting in a plasma concentration C_p. The second part, the effect compartment, models the transport of a certain plasma concentration C_p to the brain resulting in a concentration effect C_e. Finally the third part, the interaction model, describes the actual effect the drug has on the body measured by the BIS. The actual effect of the hypnotic is influenced by the effect concentration of another drug *remifentanil*, used as analgesic.
As stated in the introduction, the model can be seen as a linear dynamic part constituted by the PK model and the effect compartment, followed by a nonlinear part constituted by the interaction model. It has a so called wiener structure.
The three main parts for DoA dynamics will now be described in detail considering a particular type of modeling.

A. PK model

A three compartment model is used to emulate the patient dynamics. Compartment one represents blood and highly irrigated organs (heart, liver, kidney for example) and the other two represent the remaining organs and anatomy elements such as muscles or fat. As apparent in figure 2, compartments interact with each other, the interaction being modeled by

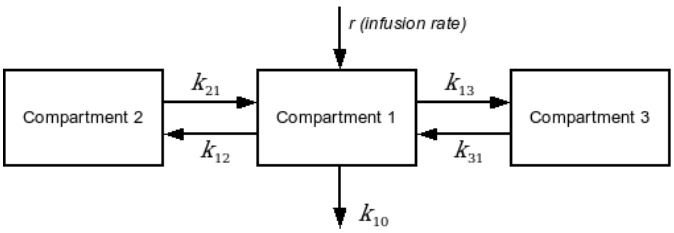

Figure 2. Three compartment PK model

the transfer coefficients $(k_{12}, k_{21}, k_{13}, k_{31})$. Coefficient k_{10} is added to model drug elimination due to metabolism.
The model is then written in a continuous state space representation [5] as

$$\begin{aligned} \dot{x} &= A_{PK}x + B_{PK}r \\ C_p &= C_{PK}x \end{aligned} \tag{1}$$

$$A_{PK} = \begin{bmatrix} -k_{10}-k_{12}-k_{13} & -k_{21} & k_{31} \\ k_{12} & -k_{21} & 0 \\ k_{13} & 0 & -k_{31} \end{bmatrix} \quad B_{PK} = \begin{bmatrix} \frac{10^4}{3600} \\ 0 \\ 0 \end{bmatrix}$$

$$C_{PK} = \begin{bmatrix} \frac{1}{1000 v_1} & 0 & 0 \end{bmatrix}$$

where v_1 represents the volume of compartment one for a given patient, and is obtained by:

$$v_1 = patw.v_c \tag{2}$$

the parameter $patw$ being the patient weight in Kilograms and v_c a coefficient which represents the volume of compartment one per unit weight.

B. Effect Compartment

The effect compartment relates the *propofol* concentration in a patient C_p with the effect concentration C_e. It is modeled by a low-pass filter [5]

$$C_e = \frac{1}{\frac{1}{k_{e0}}s + 1} C_p \tag{3}$$

or in state-space representation

$$\begin{aligned} \dot{x} &= A_{EC}x + B_{EC}C_p \\ C_e &= C_{EC}x \end{aligned} \tag{4}$$

with $A_{EC} = \begin{bmatrix} -k_{e0} \end{bmatrix} B_{EC} = \begin{bmatrix} k_{e0} \end{bmatrix} C_{EC} = 1$

C. Interaction Model

The drug interaction between *propofol* and *remifentanil* is modeled by the Hill equation relating the normalized effect concentrations U_{prop} and U_{remi} with the level of unconsciousness E

$$E = E_0 \left(1 - \frac{U_{prop} + U_{remi}}{1 + U_{prop} + U_{remi}} \right) \tag{5}$$

The normalized effect concentrations are obtained dividing each concentration by the respective concentration at half the maximum effect $U_{prop} = \frac{C_e^{prop}}{C_{50e}^{prop}}$ and $U_{remi} = \frac{C_e^{remi}}{C_{50e}^{remi}}$.The

parameter E_0 is the level of unconsciousness without drug effect.
Equation (5) shows a simplified model for the *Hill* equation. Usually an improved model is used that does not assume a purely addictive relation between U_{prop} and U_{remi}. Defining $U_{50} = 1 - \beta\theta + \beta\theta^2$ where $\theta = \frac{U_{prop}}{U_{prop}+U_{remi}}$ the improved *Hill* equation is given by:

$$E = E_0 \left(1 - \frac{[(U_{prop} + U_{remi})/U_{50}(\theta)]^\gamma}{1 + [(U_{prop} + U_{remi})/U_{50}(\theta)]^\gamma}\right) \quad (6)$$

the additional parameter γ defines the steepness of the concentration-response relation.

III. Locally weighted learning methods

The objective for this section is to describe the working principle of the LWL algorithms introduced in this paper.

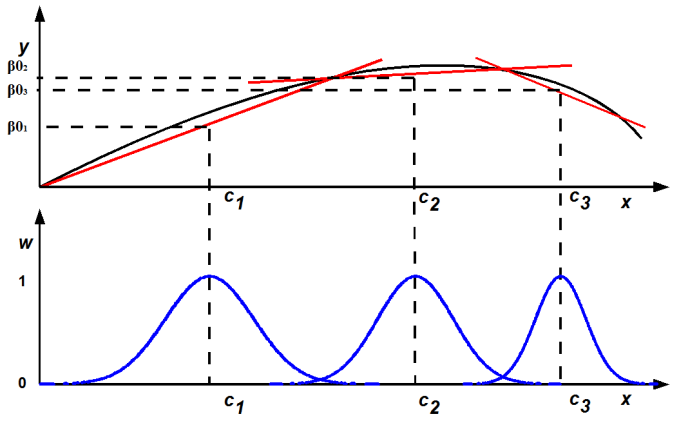

Figure 3. Linearization of a function with $D \subset \Re^n$ trough three local models (red) and their associated gaussian kernels (blue)

The main idea behind the two LWL methods is identical, consisting in approximating any given function by a variable set of locally linear functions, or models.
Consider the smooth generic function:

$$y = f(x) \quad (7)$$

where x is a vector of dimension n $[x_1 \quad \dots \quad x_n]^T$
If the domain of this function is partitioned in a sufficient number of subsets, $f(x)$ will behave linearly, or almost linearly in each one of these subsets. The approximated, or predicted value of $f(x)$ in each subset is then given by a linear function or model:

$$\hat{y}_i = \beta_i(x - c_i) + \beta_{0i} = \beta_i\tilde{x} + \beta_{0i} \quad (8)$$

where β is a vector $[\beta_1 \quad \dots \quad \beta_n]$,c_i $[c_1 \quad \dots \quad c_n]^T$ is the central point of a given section and β_{0i} is a scalar corresponding to the linear model value at the central point (see figure 3).

As apparent in the example at figure 3, in order to obtain a global predictor $\hat{y} = f(x)$ valid for any domain query point x, it is necessary to locate to which domain subset x belongs to. The associated local model is then used to make the prediction. To ascertain to which subset a query point x belongs to, LWL methods use functions denominated kernels that define each subset. Each kernel is centered around c_i and is associated to a positive definite distance matrix D_i that defines its shape. The matrix D_i defines the scope of the subset (see figure 3 for a graphical interpretation). The kernel functions used in the studied LWL methods can be of two types:

Gaussian kernel

$$w_i = e^{\frac{1}{2}(x-c_i)D_i(x-c_i)^T} \quad (9)$$

Biquadratic kernel

$$w_i = \begin{cases} (1-d^2)^2 & : \quad |d| < 1 \\ 0 & : \quad otherwise \end{cases} \quad (10)$$

$$\text{where } d = \tfrac{1}{2}(x - c_i)D_i(x - c_i)^T$$

As apparent in (9) and (10), in response to a query point x_q a kernel i gives out a weight w_i based on the relative distance of x_q to the kernel center c_i in a given direction. This relative distance can also be interpreted as the position of x_q in the subset i defined by kernel i. The further away the point x_q is from the center of the subset, the lower will be its weight w_i, having zero (Biquadratic kernel) or almost zero (Gaussian kernel) weight when x_q surpasses the boundary defined by the kernels D_i.
For further reference, a local model associated with its respective kernel will be from now on denominated as a RF and the weight obtained in a particular RF in response to a query point x_q is known as the RF activation weight for that point.
Considering the predictions obtained from each RF $\hat{y}_i$ and their associated weights w_i, the global prediction is defined as (m is the total number of *RFs*):

$$\hat{y} = \frac{\sum_{i=1}^{m} w_i\hat{y}_i}{\sum_{i=1}^{m} w_i} = \frac{\sum_{i=1}^{m} w_i(\beta_i(x - c_i) + \beta_{0i})}{\sum_{i=1}^{m} w_i} \quad (11)$$

$$= \sum_{i=1}^{m} w_i^{norm}(\beta_i(x - c_i) + \beta_{0i}) \quad (12)$$

where $w_i^{norm} = \frac{w_i}{\sum_{i=1}^{m} w_i}$ is the normalized weight with a range between 0 and 1. With $\sum_{i=1}^{m} w^{norm} = 1$.

Observing the schematic representation of (11) in figure 4 it is easier to understand the estimation process:

- A query point x_q is fed trough all RF's. In each RF an activation weight w_i is obtained by kernel i based on the relative position of x_q to its center in a given direction, the weights are then normalized.
- At the same time each local model generates a prediction based on x_q. The local predictions are then multiplied by their associated normalized weights w_i^{norm} and summed obtaining the global prediction.

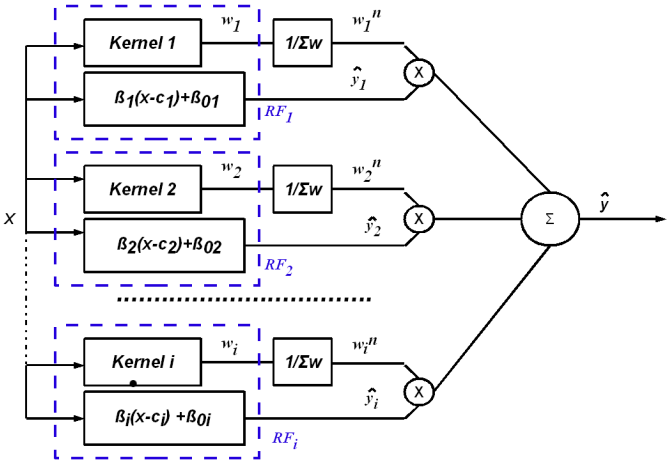

Figure 4. LWL prediction schematic

The normalized weights can be interpreted, from a prediction perspective, as a measure of relevance of a particular local prediction. The models defined in RF's to which x_q belongs the most will have greater contribution to the global prediction. The parameters (β_i, β_{0i}) for each local model are obtained trough training and independently for each RF using regression. The learning mechanism for both methods will now be described.

A. RFWR

In RFWR the regression in each RF is achieved using a weighted Recursive Least Squares Algorithm (WRLS) with an exponential forgetting factor λ. For explaining purposes the WRLS algorithm can be seen as a the online equivalent of a batch weighted Least Squares regression [8] with $p = k$ training patterns.

$$\beta = (\tilde{X}^T W \tilde{X})^{-1} \tilde{X}^T W Y = PWY \tag{13}$$

with $\tilde{X} = \begin{bmatrix}\tilde{x}^1 & \cdots & \tilde{x}^k\end{bmatrix}^T$, $\tilde{Y} = \begin{bmatrix}\tilde{y}^1 & \cdots & \tilde{y}^k\end{bmatrix}^T$ and $W = diag(w^1 \cdots w^k)$.
Looking at equation (13) it is clearer how the independent regression is achieved for each RF. From a group of mean zero training patterns $(\tilde{X}, \tilde{Y})$ for a specific RF, only the ones more relevant to it (higher w^k) will contribute to the regression.

B. LWPR

In LWPR the regression in a particular RF is performed using the a weighted incremental version of the Partial Least Squares Algorithm (PLS) [9] [10]. The mean zero patterns seen by an RF $(\tilde{X}, \tilde{Y})$ are projected into n orthogonal local directions u where univariate regression is performed. The projections u are chosen is such a way as to better explain the relation between $\tilde{X}$ and $\tilde{Y}$. As in RFWR the weights included in the regression algorithm ensure independent adaptation between RF´s.
The obtained regression parameters β_i^{PLS} are not defined in the function input-output space as in (8), but instead, correspond to the regression coefficients for each projection direction u.
Considering this fact the prediction mechanism for an RF in LWPR is slightly different from (8). When a mean zero query point $\tilde{x}_q$ is received by an RF, it is projected to each of the local projection directions u resulting in the projection vector ϕ_q the prediction is then obtained by:

$$\hat{y} = \beta_i^{PLS} \phi_q. \tag{14}$$

Another way of obtaining the prediction that is important for the proposed control strategy is to convert the parameters β_i^{PLS} to input space parameters β_i in order to obtain the prediction model in a structure similar to (8). Considering that $\beta_i = \frac{dy_i}{d\tilde{x}}$ (8) and the derivative in relation to $\tilde{x}$ for (14), the relation between β_i and β_i^{PLS} is obtained:

$$\frac{dy_i}{d\tilde{x}} = \beta_i^{PLS} \frac{d\phi}{d\tilde{x}} \Leftrightarrow \beta_i = \beta_i^{PLS} \frac{d\phi}{d\tilde{x}} \tag{15}$$

where $\frac{d\phi}{d\tilde{x}}$ is the derivative of each projection in relation to the input space.

IV. System identification using LWL methods

A causal discrete non-linear dynamic system can be approximated as a linear difference equation with time variable parameters where n is the time index:

$$y(n) = a(n)s_y + b(n)s_u \Leftrightarrow y(n) = f(s_y, s_u) \tag{16}$$

where $a(n)$ and $b(n)$ are the time variable parameter vectors and s_y s_u are the system state $s = \begin{bmatrix}s_y & s_u\end{bmatrix}^T$ components associated with the output and the input respectively:

$$s_y = \begin{bmatrix}y(n-1)y(n-2)y(n-\cdots)\end{bmatrix}$$

$$s_u = \begin{bmatrix}u(n-1)u(n-2)u(n-\cdots)\end{bmatrix}$$

Following what was stated in the general description about both LWL methods, if training patterns $[y(n), (s_y, s_u)]$ are provided, $f(s_y, s_u)$ is approximated trough m local models

$$\hat{y(n)}_i = \beta_{ai}(s_y - c_{s_y i}) + \beta_{bi}(s_u - c_{s_u i}) + \beta_{0i} \tag{17}$$

with a final prediction

$$\hat{y(n)} = \frac{\sum_{i=1}^{m} w_i(\beta_{ai}(s_y - c_{s_y i}) + \beta_{bi}(s_u - c_{s_u i}) + \beta_{0i})}{\sum_{i=1}^{m} w_i} \tag{18}$$

(17) can be interpreted as linear incremental around the point $f(c_{s_y i}, c_{s_u i}) = \beta_{0i}$

$$\Delta\hat{y(n)}_i = \beta_{ai}\Delta s_{yi} + \beta_{bi}\Delta s_{ui} \tag{19}$$

with

$$\Delta\hat{y}_i = y(n)_i - \beta_{0i}$$
$$\Delta s_{yi} = s_y - c_{s_y i}$$
$$\Delta s_{ui} = s_u - c_{s_u i}$$

substituting into (18) yields the global increment

$$\hat{\Delta y} = \frac{\sum_{i=1}^{m} w_i(\beta_{ai}\Delta s_{yi} + \beta_{bi}\Delta s_{ui})}{\sum_{i=1}^{m} w_i} \quad (20)$$

These results show that LWL methods create, from a control standpoint, a set of local linear rules of behavior (or dynamics) for the nonlinear behavior of the global system. Each one specific to a certain working area defined around (c_y, c_u). The normalized weight $w_i^{norm} = \frac{w_i}{\sum_{i=1}^{m} w_i}$ defines the contribution of a given local dynamic to the global increment. In other words w_i^{norm} measures the degree of resemblance between the global non-linear behavior and a local behavior i, when the global system is in a given working area defined by it's state (s_y, s_u).

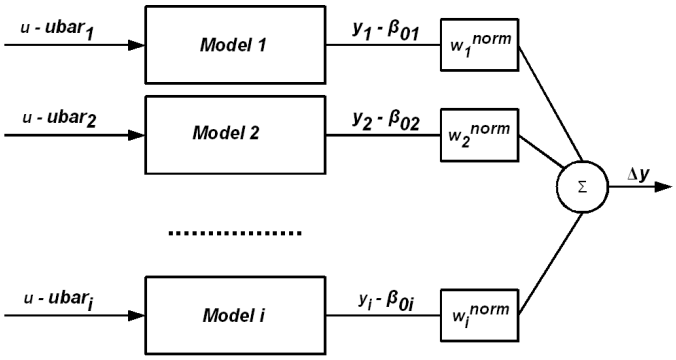

Figure 5. Schematic of the global dynamic model obtained from a LWL method

To better illustrate the concept, figure 5 shows a graphical representation of what was just said. At a given time index n, i RF's exist, each RF is a model linearized around a point $(ubar_i, \beta_{0i})$ and the final output for the global model is obtained from a linear combination of all local model's outputs. Notice the role of the normalized weights to the final output: they control which local models should be considered. Remembering that the kernel in each RF determines the weight for a current state (s_y, s_u), only the models "close" to the current state will be activated.

V. Control Strategy

Assuming each local model (19) parameters are correctly estimated and also the size and number of RFs is correctly determined (the model structure is correct). Then an augmented version of the model depicted in figure 5 can be used for control purposes. If a controller is designed for each local augmented model such that each closed loop system will track the same reference, that is:

$$lim_{n\to\infty} ref(n) - y_i(n) = 0 \Rightarrow ref(\infty) = y_i(\infty) \quad (21)$$

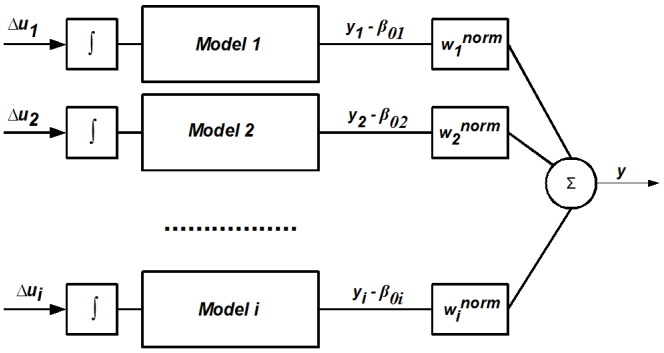

Figure 6. Augmented model for control

substituting in equation (18) it is clear the global system will also track the same reference:

$$y = \frac{\sum_{i=1}^{m} w_i ref(\infty)}{\sum_{i=1}^{m} w_i} = ref(\infty) \quad (22)$$

Notice the results shown are no proof of guaranteed convergence to the reference but instead, they were made under the rather big assumption that the global model structure and local parameters estimates were correct. Result (22) only shows the convergence of the control strategy for an ideal situation.
The speed of convergence to the reference will naturally depend on the current working area of the algorithm (each local model will have different characteristics) and also on the controller design method.

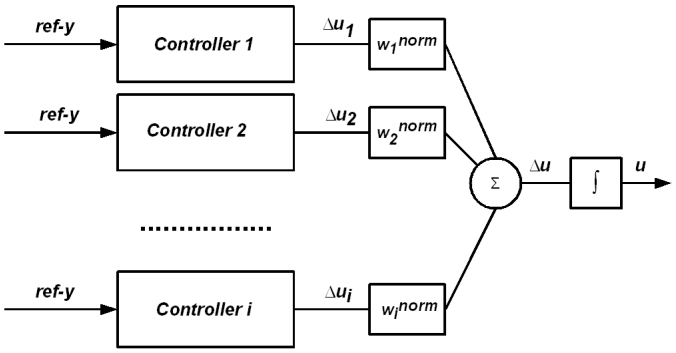

Figure 7. The controller block

The global controller will be a block of multiple controllers (figure 7), one for each linear augmented model (figure 6). The local command increments from each controller Δu_i are fused into the global increment Δu resorting to the normalized activation weights obtained from the LWL method at a given time index n. In other words, provided that the global system state at a time n, the most influential controllers to Δu are those that control the most activated local models.
The remainder of this section will describe how each controller

is derived from the respective local model, together with a description of the control algorithm.

A. Local controller synthesis

To synthesize each local controller it is necessary to convert each local model defined by

$$y_i(n) = \beta_a s_y + \beta_b s_u \tag{23}$$

into a State-Space formulation.

$$\begin{aligned} x(n+1) &= \Phi x(n) + \Gamma u(n) \\ y(n) &= Cx(n) \end{aligned} \tag{24}$$

This is accomplished by making

$$\Phi = \begin{bmatrix} 0_{1XN_a-1} & & I_{N_a-1} \\ \beta_{a_{N_a}} & \cdots & \beta_{a_1} \end{bmatrix} \Gamma = \begin{bmatrix} 0_{1XN_a-1} \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0_{dX1} & \beta_{b_{N_b}} \cdots \beta_{b_1} \end{bmatrix}$$

where 0_{nXm} is a n by m matrix of zeros and I_n is a size n identity matrix.

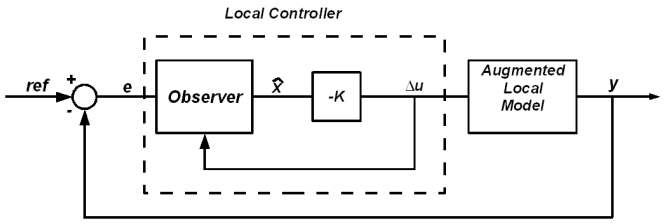

Figure 8. Local control scheme

Using the dynamics described by (24) a controller like the one specified in figure 8 is designed. An *Euler* integrator $\frac{Ts}{z-1}$ (T_s is the discrete sampling time)is added to the the local model, leading to the augmented dynamic:

$$\Phi_a = \begin{bmatrix} \Phi & \Gamma Ts \\ 0 & 1 \end{bmatrix} \Gamma_a = \begin{bmatrix} 0 \\ Ts \end{bmatrix} C_a = \begin{bmatrix} C & 0 \end{bmatrix}$$

The control law gains K are determined for the augmented system solving a discrete infinite horizon LQ optimization problem [11] [12]

$$J(u) = \sum_{n=1}^{\infty} [x(n)^T Q x(n) + u(n)^T R u(n)] \tag{25}$$

where Q is square matrix penalizing large sates x and R is a scalar penalizing large commands u. This problem has a closed form solution S obtained from the discrete algebraic *Riccati* equation

$$\Phi_a^T S \Phi_a - S - (\Phi_a^T S \Gamma_a)(\Gamma_a^T S \Gamma_a + R)^{-1}(\Gamma_a^T S \Phi_a) + Q = 0 \tag{26}$$

The control law gains K that stabilize the closed loop system are then computed from S by:

$$K = (\Gamma_a^T S \Gamma_a + R)^{-1})(\Gamma_a^T S \Phi_a) \tag{27}$$

To obtain the sate estimate $\hat{x}$ a current observer [11] incorporated with the system's tracking error $e(n) = ref(n) - y(n)$ is used:

$$\begin{aligned} \hat{x}(n|n) = \hat{x}(n|n-1) + L_c[-e(n) - C_a \hat{x}(n|n-1)] \Leftrightarrow \\ \Leftrightarrow \hat{x}(n|n) = [I - L_c]\hat{x}(n|n-1) - L_c e(n) \end{aligned} \tag{28}$$

where I is the identity L_c are the current observer gains and $x(n|n-1)$ is the predicted estimate based on a model prediction from the previous time estimate

$$\hat{x}(n|n-1) = \Phi_a \hat{x}(n-1|n-1) + \Gamma_a u(n-1) \tag{29}$$

The observer gains L_c are designed to make the observer a factor faster than the closed-loop system. This is done by placing the observer poles Op at a location f times smaller than the closed loop poles CLp

$$Op = CLp^f \tag{30}$$

with $f > 1$.
Based on (29) the current controller can then be defined by

$$\begin{aligned} \hat{x}(n+1|n) &= [\Phi_a - \Gamma_a K]\hat{x}(n|n) \\ \Delta u &= -K\hat{x}(n|n) \end{aligned} \tag{31}$$

An implementable controller is then derived substituting (28) into (31)

$$\begin{aligned} \hat{x}(n+1|n) &= [\Phi_a - \Gamma_a K][I - L_c C_a]\hat{x}(n|n-1) \\ &\quad -[\Phi_a - \Gamma_a K]L_c e(n) \\ \Delta u &= -K[I - L_c C_a]\hat{x}(n|n-1) + K L_c e(n) \end{aligned} \tag{32}$$

B. Control algorithm

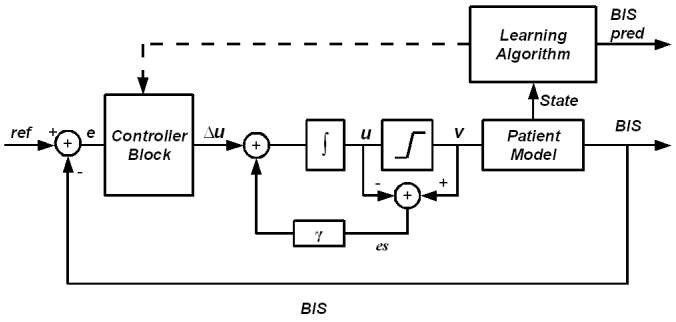

Figure 9. Overall view of the control strategy. The controller block synchronized with the learning algorithm and the common integrator with anti wind-up

The algorithm is designed to work with one of the two methods described in this work RFWR or LWPR and can work with a pre-trained initial mean model or without any prior information. It comprises of two main stages:

- In the first stage the system is being controlled manually and training patterns $[y, (s_y, s_u)]$ are being supplied to the learning algorithm at each time index n. In this stage the algorithm either builds an initial model for the case

where no a priory information was known, or uses the new data to adjust the pre-trained mean model.
- In the second stage control is turned on based on the initial model obtained in the first stage. Training patterns continue to be supplied to the learning algorithm and the control algorithm checks for changes in the global model, the controller block is then updated accordingly.

1) Updating the Controller Block: The updates for the controller block are done periodically with a period T defined in the initialization. During the update process three distinct situations may occur:

- A new RF is added, in this case a controller is synthesized for the new model and added to the controller block.
- The model parameters for a pre-existing RF change, in this case a controller for the new model is synthesized and the controller corresponding to the RF in question is substituted. To avoid constant changes in the controller block caused by small changes in the model *Vinicombe* gap metric [13] is used to compare the pre-existing model with the new one. The controller only changes if the gap metric between the two models is superior to a given threshold.
- An RF is pruned, in this case the controller associated to the RF in question is eliminated from the controller block.

2) Anti-windup: It was also necessary to implement an anti-windup strategy for the controller block common integrator (see figure 9). This was done because the actuator used for DoA is a computer controlled syringe that has a saturation at 0, thus constituting a nonlinear actuator.
The strategy chosen for the anti-windup was back-calculation. The working principle for this strategy is as follows:

- When the actuator is not saturated $u = v$ and thus $es = 0$, meaning the integrator works normally.
- When saturation is reached $es = u - v$ and the integrator input is driven to zero at a rate γ with $\gamma >= 1$.

VI. Results & Conclusions

The proposed objective was to develop a control strategy using a LWL method to identify a patients dynamics. The basic assumption was that if a good prediction model was obtained from the LWL method, it would represent the dynamic behavior of the real system and a controller based on the predictive model could be used.
Both learning methods (LWPR and RFWR) were tested in terms of prediction performance in a comparative analysis using different initial parameter configurations for both methods. Despite having a higher prediction error, LWPR, proved to be more reliable. It presented less prediction failures (see figure 10) due to its PLS implementation. And was thus considered more suitable for control.
The best predictive models obtained from LWPR were tested in terms of control performance using the DoA models of 8 test patients as control plant. It was verified high control performance variation exists between LWPR predictive

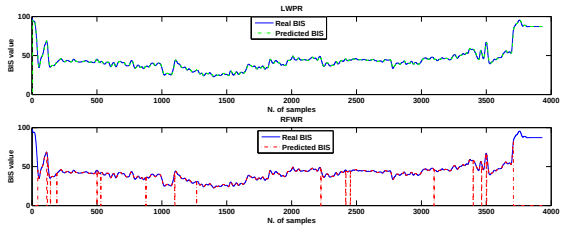

Figure 10. Prediction performance test result, LWPR vs RFWR. RFWR fails predictions.

models obtained with different parameter configurations. As illustrated in figure 11 the prediction performance alone does not provide guarantee of adequate control with this strategy.

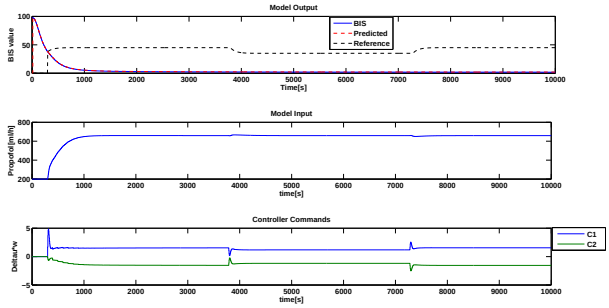

Figure 11. Simulation result example where prediction performance is good and yet control fails completely.

However some of the tested predictive models, were able to control each of the 8 test patients under the influence of noise (figure 14) in a wide working area with satisfactory results. Figure 12 shows a simulation using a particular

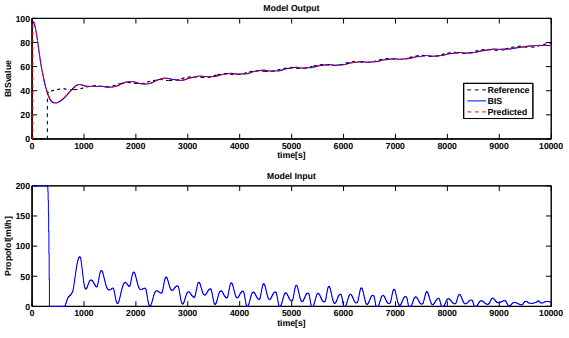

Figure 12. Simulation result for a wide range reference, system input and output.

predictive model. The reference is set as a disturbed ramp ranging between BIS levels 40 and 70. The objective for this simulation was to evaluate the controller behavior when the system changed working area, figure 13 shows the controller

block adaptation to the working area. In the beginning of simulation local controllers 3 and 4, associated with the local predictive models defined in that initial working area, are more influential for control. As the working area changes the most active controllers change accordingly.

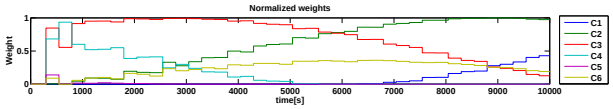

Figure 13. Weights associated with each local controller for wide range reference simulation.

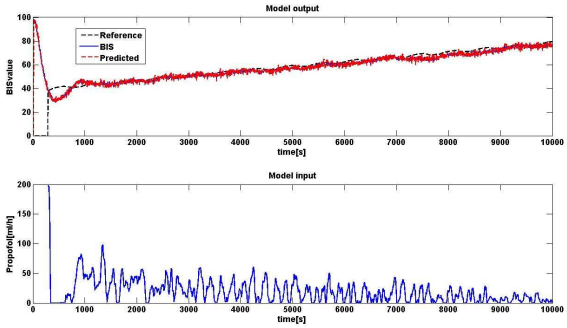

Figure 14. Simulation result for a wide range reference with sensor noise added, system input and output.

The use of online adaptation to adjust the initial controller block is still an open subject. One of the predictive models was tested using online adaptation. In a situation without noise the novel data integrated in LWPR did not cause any change in the controller block. Adding new data corrupted by noise caused control to fail (see figure 15).

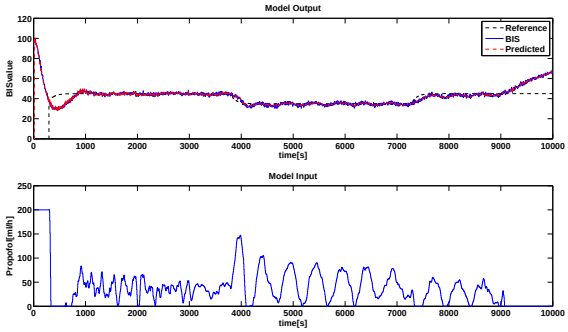

Figure 15. Plant input and output with gaussian noise $\sigma = 1$

REFERENCES

[1] C. G. Schaal, S.Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, 1998.

[2] S. Vijayakumar and S. Schaal, "An o(n) algorithm for incremental real time learning in high dimensional space," *International Conference on Machine Learning, Proceedings of the Sixteenth Conference*, 2000.

[3] S. S. S. Vijayakumar, A. DSouza, "Incremental online learning in high dimensions," *Neural Computation*, 2005.

[4] M. H. G. A. D. S. Bibiany, C. R. Ries, "Clinical anesthesia and control engineering: Terminology, concepts and issues," *EUROPEAN CONTROL CONFERENCE,*, 2003.

[5] N. C. D. Castro, "A simulink compartmental model for depth of anaesthesia," INESC-ID (Portugal) on leave from INPGrenoble - ESISAR (France), Tech. Rep., April 2008.

[6] B. M. W. M. N. Marsh and G. N. C. Kenny, "Pharmacokinetic model driven infusion of propofol in children," *Bristish Journal of Anaesthesia*, 1991.

[7] K. S. Stadler, "Modelling and control in anaesthesia from design to validation." Ph.D. dissertation, ETH Zuerich, 2003.

[8] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*, M. Cambridge, Ed. MIT Press, 1986.

[9] H. Wold, "Soft modeling by latent variables: the nonlinear iterative partial least squares approach." *Perspectives in Probability and Statistics,*, 1975.

[10] J. Frank, I.E. Friedman, "statistical view of some chemometric regression tools." *Technometrics*, 1993.

[11] M. L. W. Gene F. Franklin, J. David Powell, *Digital Control of Dynamic Systems*, 3rd ed. Addison-Wesley, 1998.

[12] B. W. Karl J.Åström, *Computer Controlled Systems - Theory and Design*, 3rd ed. Prentice Hall, 1997.

[13] G. Vinnicombe, "Frequency domain uncertainty and the graph topology," *IEEE Transactions on Automatic Control*, 1993.