LIFE-LIKE CONTROL

Heikki Hyötyniemi
Helsinki University of Technology
Control Engineering Laboratory
P.O. Box 5400, FIN-02015 HUT, Finland

Typically when proposing cognitive architectures, one is exclusively interested in the analysis of (sensory) input signals, and the construction of the (motor) output signals is ignored. In this paper, a framework that was also designed for one-way operation is extended to carry out two-way signal processing. It turns out that systems theoretically well motivated control structures for high-dimensional, nonlinear processes can readily be implemented.

1 INTRODUCTION

In the fields of artificial intelligence and cognitive science, new architectures and fancy ideas are introduced every now and then. It is difficult to compare different architectures — selecting a different point of view, and defining "goodness" in a special way, almost any idea can be deemed as the best. Could one find such criteria that could be agreed upon also by the cognitive scientist often having a very non-technical background?

Introducing new degrees of freedom, practically any architecture can explain observations. From the technical modeling point of view, *simplicity* is a good validity criterion (in the spirit of *Ockham's razor*). However, as an only starting point, the simplicity goal is not truly convincing: Who has ever said (Stephen Wolfram does not qualify!) that the extremely complicated tasks carried out by the mental machinery ever *should* be solved by simplistic approaches? It seems that humanists feel hurt when the human excellence is underrated in such a shameless way.

Another heuristic criterion that could perhaps be exploited is *intuitive appeal.* How easily an architecture that has been designed for a special cognitive task can be modified to solve a totally different task?[1] For example, it is a tradition to propose architectures for perception, for analysis of input signals — the question that can be asked is whether these architectures can act as frameworks for *control,* for construction of output signals. This is an opposite but equally relevant mental task. Specially, can the approaches to *life-like perception* be extended to *life-like control?*

In this paper, approaches to "intelligent control" are first discussed, and then, the new cognitively motivated controller structure is presented. This controller scheme is applied to control of a simulated robot arm.

[1]In [6], a different approach to "intuitive plausibility" was studied: A cognitive model is "good" if the *errors* it makes after its capacity has been exceeded are somehow "expert-like"

2 COGNITION AND CONTROL

2.1 Controlling of systems

To a layman, a well-tuned control system looks like an intelligent agent, appropriately reacting to the system behavior and compensating for deviations from desired values. Again, knowing how it is done, destroys all magic: The control signal is simply calculated according to some fixed function. The actual intelligence is buried in the construction of that function, and the understanding remains in the head of the control engineer.

On the higher (more intelligent?) level, different kinds of *adaptive control* strategies have been proposed [1]; lately, new approaches like *iterative learning control (ILC)* [14] have increasingly gained interest. What is common to all these adaptation schemes is that they are typically based on the differentiability properties of a cost criterion; they are often based on some kind of identification of system parameters, either applying gradient algorithm or some more efficient method. From the cognitive point of view, there are plausibility problems here. Traditional adaptive control schemes update the control strategy gradually, whereas, when lifting some object, for example, it only takes a fraction of a second for a human to adapt and apply appropriate forces, no matter what is the weight of that object.

However, there are plenty of conceptual tools and other intuitions that are provided to us by the modern control theory when thinking of "intelligent" control structures:

- One can be confident that cognitive, representation-based control, as opposed to "behavioristic" control, is feasible: The theory says that it is possible to divide the controller construction task into two parts — first the "input-oriented" state estimation, and, second, the actual "output-oriented" control signal construction — without jeopardizing something essential. This means that it need not necessarily be a straightforward input–output structure that is necessary for control applications, but an internal structure is equally (or even better?) motivated.
- These separate tasks of state estimation and control construction are *dual phenomena,* where the essentially same principles and tools (solving of Riccati equations, for example) can be applied to construct optimal estimation and optimal control strategies alike. This duality principle can perhaps be extended to cognitive control: It might be possible that the extensively studied perception mechanisms could also be applied in the opposite way.

2.2 AI approaches

There are different approaches to integrating intuitions acquired in the fields of artificial intelligence and cognitive science to control of systems. In Fig. 1, an illustration of the different prototypical intelligent control strategies is shown. Different approaches are projected onto two axes: *Physical fidelity* here means that the controller structure has some neural plausibility, and there is some connection to the underlying *quantitative* building blocks; *functional fidelity,* on the other hand, represents whether some cognitively relevant functionalities can be implemented, and whether it is possible to construct some *qualitative* structures in that framework. Evidently, traditional control (TC) has none of these properties; (feedforward) neural network control (NC) implements the neural ideas carefully, but no structures emerge from the mass of parameters; using expert system

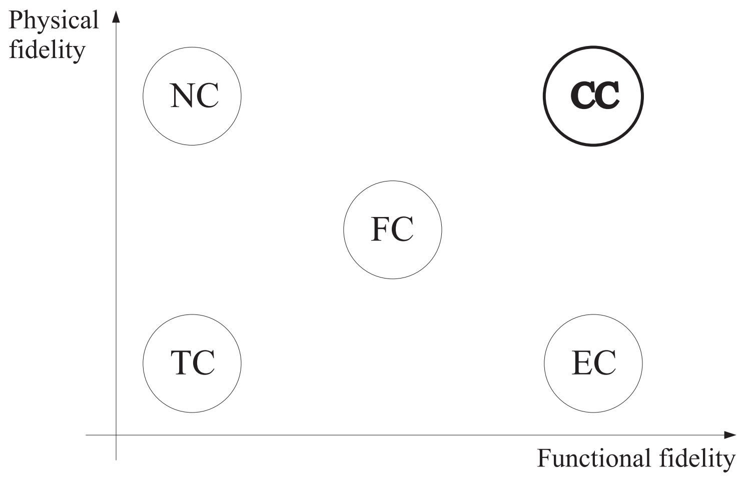

Figure 1: Schematic illustration of the properties of different "intelligent" control strategies (see text)

based control (EC), cognitively non-trivial inference rules can be implemented, but these reasoning structures are purely symbolic. Fuzzy control (FC) was explicitly defined to solve these isolation problems of EC's; perhaps it is natural that these fuzzy approaches where nothing is binary fall somewhere in between straightforward characterizations — there are rules, yes, but after all they are collapsed into simple black-box functions; and even though they are numeric and there is a close connection to actual measurements, they cannot be adapted in a natural way.

It is claimed here that the *chunk control* (CC) that will be elaborated on below is a nice compromize what comes to these two cognitive plausibility criteria: The controller can adapt according to observations in a rather plausible way even if the system is complex and high-dimensional; further, these learned data structures constitute representations that can be claimed to have also some fundamental cognitive plausibility.

2.3 New view

In [7] it was assumed that cognitive phenomena could be attacked — if the view of *naturalistic* and *contextual semantics* is adopted — in a a framework of multivariate statistics. There are dependency structures among the information atoms in the high-dimensional data space, and one only needs to model these dependencies to have some "intelligent-looking" behavior emerge. To make this modeling task feasible, there must exist some natural structural principles what comes to the data distributions. It was assumed that data is *clustered,* and within these clusters there is some internal structure; this internal structure consists of local *linear subspaces* determining the cluster "shape". Statistically, one cluster represents a single unimodal (Gaussian) subdistribution in a mixture model, meaning that, in the maximum likelihood sense, the local dependencies between variables can be modeled using linear functions. Depending at what level one is studying the cognitive phenomena, the clusters can be called either *categories, concepts,* or *patterns,* and the subspace axes are either *attributes, chunks,* or *features.*

The "patterns" and "features" are applicable on the lowest level, where actual measurement signals (or sensations) are being processed. On the other hand, when studying control and manipulation of systems, one is again operating directly on very concrete signals. However, even though one is operating on the same signals, it is the point of view and the concepts that are employed that efficiently dictate what kind of things

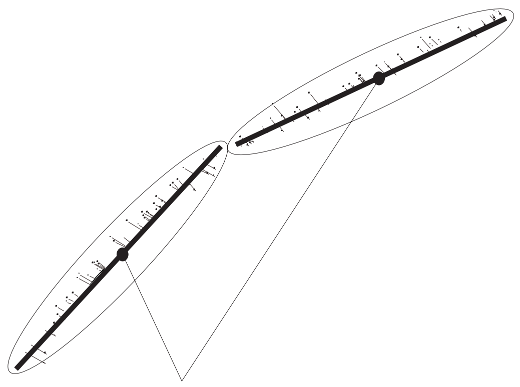

Figure 2: Idea — categories stand for operating points, attributes for degrees of freedom. Smooth nonlinearities can be approximated using piecewise linear representations

one can think of (remember Wittgenstein!). The terms like "pattern" and "feature" are conceptually too tightly coupled to one-way processing of input modalities and they are semantically too loaded to help us having intuition in the cognitive control problem.

Indeed, modeling involves introducing new abstractions and concepts. However, here it seems that we are lucky: Very few new concepts need to be introduced. It turns out that in the field of *system theory* and *control engineering* ready-to-use concepts already exist. It is only a question of renaming the cognitive concepts, and slightly extending the traditional view of control engineering concepts (see Fig. 2):

- The cluster centers (categories, patterns) are now *operating points,* around which the system can be locally linearized.
- The subspace axes (chunks, features) are now *degrees of freedom,* determining the local linear dependencies between variables.

This means that there are various operating regimes around the space, located in regions that are relevant to the system. As the system traverses in the state space it every now and then moves from a regime to another, and the feedback control law changes accordingly. In concrete terms, the behavior of the controller is *piecewise linear.* However, there are various theoretical challenges facing us when implementing this control strategy.

3 CHALLENGES

3.1 Input–output structure

When studying the problem of extending the one-way perception approaches towards two-way control, the first obstacle concerns the basic structure: The model needs to be extended to include not only input signals but also output signals. A general framework for such causal input–output mappings is *regression.* In concrete terms, it is now assumed that the input vector is denoted y, and the regression model should produce u, the output vector[2].

In fact, regression problems have been extensively studied also in the neural networks community, no doubt because they are so useful in different kinds of practical applications. For example, the feedforward neural networks *are* regression structures between the input

[2]Note the inversion of causalities: Normally in control engineering, u is the input, and y is the output; however, now one is studying the feedback loop, where one should construct an appropriate system input, the control signal u as a function of the observed system output y

layer and the output layer. The actual problem here is that such "cognitive regression" strategies easily become "behavioristic" black box models with the internal structure having minor role.

There exists activity also for extending the exclusively input-oriented neural network schemes: For example, *self-organizing maps* have been used as a framework for regression. Whereas the direct SOM-based regression is discrete-valued, continuous output can be reached by introducing additional operations ("parameterized SOM"), or if some local regression models are explicitly implemented in the nodes. However, in these cases the basic SOM structure has been extended considerably. In the same way, a straightforward approach in our case would be to match the observation against the internal model, receiving the "perception vector" x (see [10]), and explicitly construct a mapping from x to u using, for example, least squares matching. From the point of view of cognitive plausibility, there are some problems that would be faced:

- The first argument concerns added complexity: Yet another layer needs to be added on top of the basic structure, and new parameter adaptation schemes need to be introduced. Even though simplicity cannot be the only guiding principle when struggling towards cognitive plausibility, it would be a nice bonus.
- The second argument is more acute: One should fix the input–output structure already during the model construction phase. It is a well-known fact, however, that causalities cannot be seen in the data (remember Hume) — this means that the inputs and outputs should be explicitly preprogrammed by some external mind.

The approach that will now be concentrated on has *no separate outputs,* but inputs and outputs are processed similarly. The roles of inputs and outputs become distinguished only through their usage: If some signals are known at some time, they are regarded as inputs at that time; if they are not known they are reconstructed based on the learned internal dependency model between the signals, and these signals are regarded as outputs at that time. This means that the actual augmented data vector is

$$f = \left(\frac{y}{u} \right). \tag{1}$$

It is assumed that the statistical properties of such vectors are modeled as presented, for example, in [7], resulting in a set of feature vectors (chunks) collected in the matrix C, containing some kind of correlation structures between the variables. As contrasted to the assumptions made in [10], the model is now *clusterwise linear,* no "f_{cut}" function is needed now. If the dimension of the input y is n, and the dimension of the output u is m, the dimension of the combined data is $n + m$. The portions of C standing for the input block and the output block can formally be restored as

$$C_y = \left(\begin{array}{c|c} I_n & \mathbf{0} \end{array} \right) \cdot C \qquad \text{and} \qquad C_u = \left(\begin{array}{c|c} \mathbf{0} & I_m \end{array} \right) \cdot C. \tag{2}$$

Now, let us define *associative regression* as a structure where the known quantities in y are first used to determine the internal image x, and only after that, the estimate for the unknown variables is constructed based on this x (see Figs. 3 and 4). Using a diagonal square weighting matrix W_y that only emphasizes the variables in y, it is also possible to express the associative matching directly as a weighted least-squares solution

$$x = \left(C^T W_y C \right)^{-1} C^T W_y \cdot f. \tag{3}$$

Associative regression

1. Find the best possible match between the model and the known variables in y applying the least-squares (pseudoinverse) formula, not taking into account the unknown ones in u:

$$x = \left(C_y^T C_y\right)^{-1} C_y^T \cdot y. \tag{4}$$

2. After x is found, the reconstruction of the output variables u is implemented by calculating how the stored model structure reconstructs those variables when x is fixed:

$$\hat{u} = C_u \cdot x = C_u \left(C_y^T C_y\right)^{-1} C_y^T \cdot y. \tag{5}$$

It needs to be recognized that whereas "association" is traditionally associated with discrete-valued recall, that is, typical associative memories can only store a finite number of distinct patterns, now the output u is a continuous function of y, so that there is an infinite number of possible variable combinations. It is assumed that some compression takes place during regression, so that x is lower-dimensional than y, hopefully containing the non-redundant noise-filtered information of the measurement, and $C_y^T C_y$ is invertible.

There are some nice characteristics about the proposed regression idea: One of them is that it is naturally a MIMO structure (multi-input, multi-output). Note that the coherence between input and output signals may also facilitate other applications in addition to regression and control. For example, if it happens to be some of the input signals that is not accurately known, one can use the model for reconstructing (filtering, smoothing) of the uncertain signal values:

$$\hat{y} = C_y \cdot x = C_y \left(C_y^T C_y\right)^{-1} C_y^T \cdot y. \tag{6}$$

Whether or not this associative recall actually works as proposed, is very much dependent of the construction of the C matrix; this issue is elaborated on in the next section.

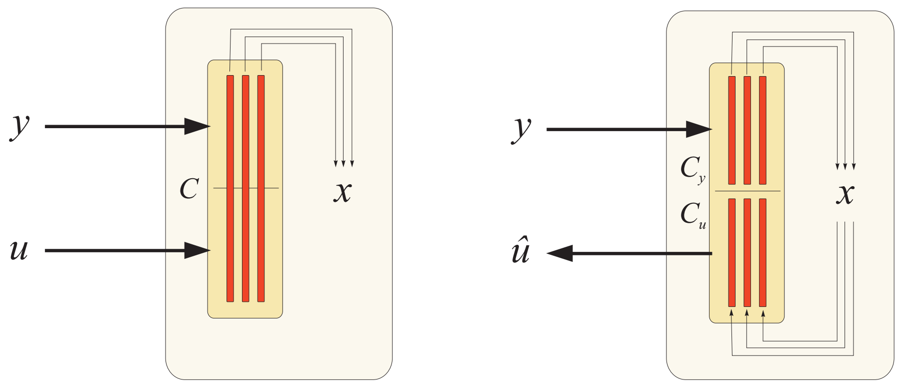

Figure 3: "Associative regression" — learning the correspondences between the signals, on the left, and applying the model, on the right

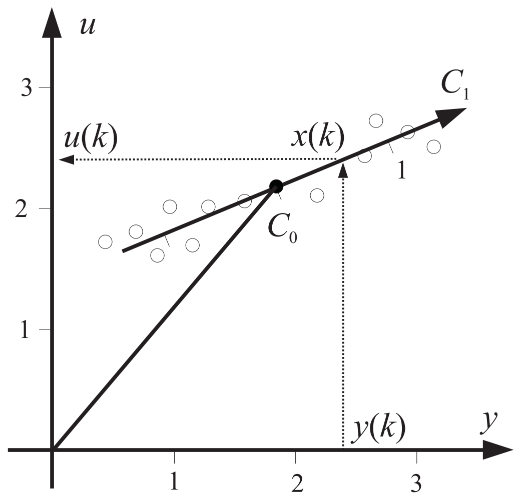

Figure 4: Illustration of the operation of the associative regression in one dimension, that is, $n = 1$ and $m = 1$. The value of y is first projected onto some kind of correlation structure between y and u (typically a set of *principal* or *independent components*), and only after that the internal latent variable x is projected onto u

3.2 Uncertainty

Many cognitive models remain on the purely symbolic level. Now, however, one is facing the challenge of real-life data in two ways: First, the perceptions are to be extracted from the available data, and, second, the results are to be transformed back to real life signals. The architecture has to be robust against the real data problems like noise and high dimensionality.

The role of the internal model is to capture the statistical relationships among the data, and these correlation structures are represented as columns in the matrix C. In the basic case, these structures are the *principal components* spanning the variance structure in the data space. Assuming that the data distribution is unimodal and Gaussian, the linear principal component structure *optimally* captures the observed correlations among the data. Data reduction is achieved and (hopefully) only the relevant information is captured when the data is projected onto the most significant principal components; the high-dimensional noisy y is represented by the low-dimensional x.

Principal components are routinely used as a basis for *structured* regression approaches. When the input data is first projected onto the subspace spanned by a subset of principal components, and from there further to the output, one has the well-known statistical regression method *principal component regression (PCR).* What is different in associative regression is that these principal components are determined *simultaneously* for y and u; in this sense, one could assume that the problem sometimes experienced with PCR — only emphasizing input data — could now be avoided (see [8]).

3.3 Dynamic nature

Representing dynamics or modeling time-dependent phenomena are typically not worried about in cognitive architectures. However, as revealed by systems theory, there are three phenomena that make the control applications in the presented framework specially attractive:

1. System dynamics can be captured in the (possibly high-dimensional) state vector.

2. Optimal (in a certain sense) control for a linear (affine) system is linear (affine).

3. Optimization can be carried out in a modular manner.

The theory says (for example, see [2]) that the dynamics of an N'th order linear differential equation can be captured in an N dimensional *state vector,* meaning that if the state is known, all history can be forgotten. When looking at the proposed cognitive structure, it is the vector x that now stands for the state; in fact, the vector y can also be regarded as non-minimal state, whereas it is assumed that in x this information is compressed (note that the state is not unique).

Assuming that the system behavior is smooth, the nonlinearities can locally be approximated by *affine* models, meaning that in addition to the linear part, there is a constant drift term in the model. The theory also says that if the system is affine, and if the cost criterion is quadratic (see later), the control law optimizing the criterion also is affine. And it is affine models that can exactly be implemented in the proposed framework. This means that for linear systems, truly optimal control can be implemented using chunk control; if the system can only be piecewise linearized, the result is suboptimal. The more there are local affine submodels being employed, the better the approximation should become.

Yet another conceptual tool is needed to make the cognitive control structure truly functional: This is based on the idea of *dynamic programming* and *principle of optimality.* Assuming that the system is time-invariant (given the same inputs, the system behaves the same way no matter what is the actual time point), one can divide an optimization problem in parts. When one first determines the optimal control law in the vicinity of the goal state, it is only needed to optimally transfer the system into the basin of this already optimized submodel — and the control along the whole trajectory becomes optimal! This principle is applicable also to nonlinear systems.

3.4 **Adaptivity**

When the system model has been learned, its parameters should not be changed carelessly — otherwise the acquired information about the system behavior is soon lost. How could we then reach naturally plausible, very fast and flexible adaptation of the closed-loop system behavior?

Remember that it was assumed that the dimension of the observation y can be very high, containing perhaps also some measurements that on the lowest level seemingly do not belong to the dynamic model. When control based on associative regression is implemented, as presented later, its robustness against redundancy makes it possible to reach "life-like adaptive control": For example, it is possible to include in y information about the environment (say, sensation of the load weight), and if it turns out that this quantity correlates with other variables, changing the system dynamics, its contribution is automatically integrated in the model (assuming that this contribution is continuous and locally linear). This facilitates extremely fast system adaptation according to environmental disturbances.

What comes to traditional adaptive control approaches (see [1]), it can be noted that the resulting control takes the next step beyond *gain scheduling,* and provides a new framework for *multi-model control:* Different operating regimes are selected according to the observed situation, resembling also *switched control* that is used specially in hybrid systems. Indeed, now there are *two* levels of adaptation: The slow adaptation varies the controller parameters, whereas the fast adaptation is gain scheduling, selecting the appropriate submodel and constructing the control law accordingly (see [9]).

4 "CHUNK CONTROL"

4.1 Learning by being shown

The basic idea when implementing control based on the discussions above is that such vectors f are stored in the model that correspond to intended controller behavior. That is, the vector f is constructed by combining the observed measurement y and the control signal u given by the controller to be emulated. During run-time, u is associatively extracted when y alone is given. There are many names that could be given to such cognitive control strategy; functionally, it is *associative control,* and physically it is *feature-based control* or *chunk control.*

A single local model is represented as a matrix C of dimension $(m+n) \times (N+1)$, so that the first column C_0 represents the center point, and the other N vectors C_1^c to C_N^c stand for the assumed N degrees of freedom within the operating region[3]:

$$C = \left(\begin{array}{c|c|c} C_0 & \cdots & C_N \end{array} \right). \tag{7}$$

In principle, a single model C suffices if the system is linear enough; however, typically more of them are needed, one for each operating point. The model for operating point c is denoted C^c.

The iterative learning of the principal components feature model can be based on, for example, *Generalized Hebbian Algorithm (GHA)* originally developed by Professor Erkki Oja. The learning is carried out separately in each submodel, and the data can be assumed to be unimodal — that is why the *Generalized GHA* [5] is not needed. As compared to traditional principal component extraction procedures, one now needs to recognize that the data is not zero-mean, and this mean vector (operating point center) needs to be processed in a special way. During the learning, the following procedure is repeated as long as needed:

1. Construct a new observation vector f:

$$f = \left(\frac{y}{u} \right). \tag{8}$$

2. Match f against all submodels to find that with the best fit:

$$\bar{c} = \arg\min_c \left\{ \| f - C_0^c \| \right\}. \tag{9}$$

3. Apply the modified GHA algorithm for all submodels c, for all i from 0 to N:

$$\left\{ \begin{array}{lcll} F_i & = & F_{i-1} - z_{i-1} \cdot C_{i-1}, \quad F_0 = f & \text{Input to layer } i \\ z_i & = & F_i^T \cdot C_i, \quad z_0 = 1 & \text{Correlation (unscaled)} \\ C_i & \leftarrow & C_i + \gamma h \cdot z_i \cdot (F_i - z_i \cdot C_i) & \text{Chunk adaptation} \end{array} \right. \tag{10}$$

[3]From now on, subindices refer to matrix columns or single vector elements, whichever is appropriate; capital letters normally denote matrices and lowercase letters vectors and scalars

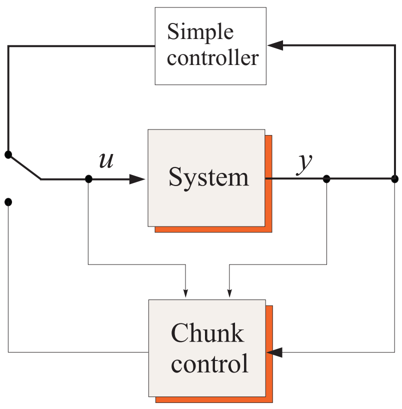

Figure 5: First, a stabilizing control is seen, and after that, the same behavior can be reproduced

In Step 2, the best matching submodel can also be selected using more sophisticated criteria, emphasizing the actual outlook of the data clusters, and employing the covariance matrices (see [10]):

$$\bar{c} = \arg\min_{c} \left\{ \|(f - C_0^c) - \left(\begin{array}{c|c|c} C_1^c & \cdots & C_N^c \end{array} \right) \cdot x\|_{\Sigma_e^{-1}} + \|x\|_{\Sigma_x^{-1}} \right\}. \tag{11}$$

In Step 3, when implementing the GHA algorithm, F denotes the matrix of "virtual" inputs where contributions of the previous levels have been eliminated, and γ is the adaptation step size. A few modifications are here needed as compared to the traditional GHA: First, z^0 (the first-level unscaled correlation) is set to 1 to take into account the special role of C_0 as a non-scaled center vector; second, the neighborhood effect $h = h_{\bar{c},c}$ is applied to implement self-organization among models (see [13]), and to keep all submodels C^c involved. It seems that the convergence of GHA is typically a rather time-consuming process; however, the main thing is that the input and output variables in C's are in balance defining an appropriate subspace among the variables. It does not matter so much if the axis directions have not yet converged to the final principal components.

When training is being carried out, various cycles of transient behaviors are run in succession, starting from some initial state and ending in the goal state, applying some available stabilizing controller (see Figs. 5, 6, and 7). The feature model adapts to the appropriate signals so that after training more or less identical controls can be restored using associative regression.

Variation of starting state is needed to introduce some fresh information in the training data. Problems may emerge if there does not exist enough excitation in the system; if the model structure is too complicated as compared to the experienced behaviors, not all degrees of freedom are necessarily spanned by the observation data.

4.2 Optimization by trial-and-error

When thinking of model adaptation and optimization of behavior, it would be tempting to introduce some higher-level "intelligent agent" that could carry out this task. However, this kind of mysterious higher levels should be avoided to keep things simple — how to reach plausible "life-like" operation also here?

The idea applied here is to run the existing learned control strategy various times, every now and then slightly modifying the control from the nominal values as determined by the chunk model. If the behavior is enhanced as compared to the original behavior, the

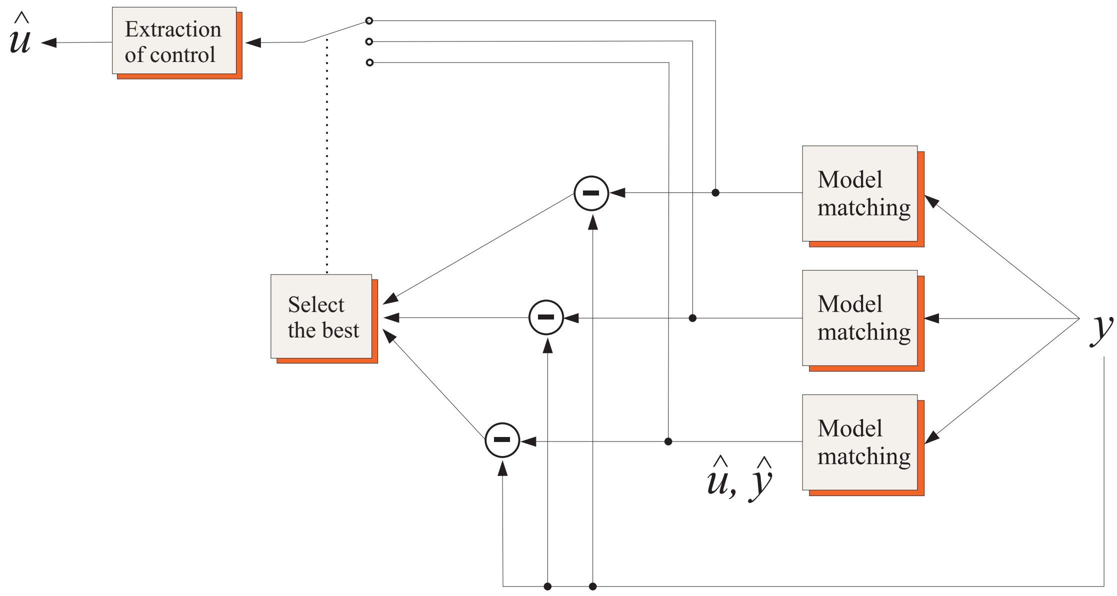

Figure 6: Competition for control among local linear submodels. Each of the submodels tries to explain the observation y as accurately as possible; the best of them is selected for control construction

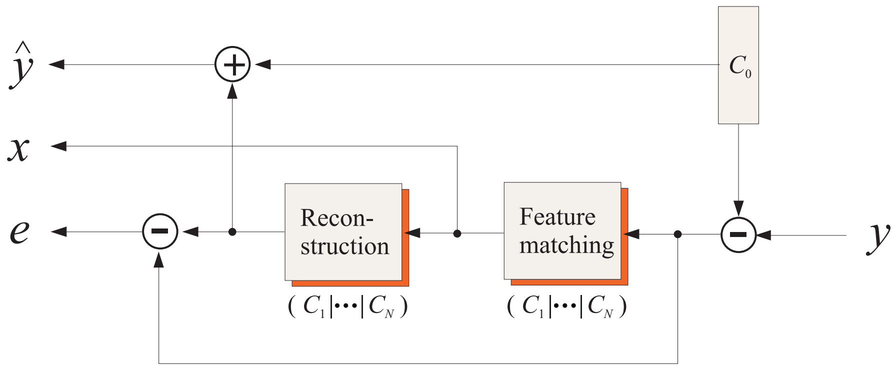

Figure 7: Contents of a single model matching block. Special kind of feature matching is needed because of the center point C_0

data f with the modified control is learned as presented before. One only needs to have some outside *critic* saying whether the behavior is “good” or if it is “bad”.

This kind of random search is not so hopeless as it may sound: First, assuming that the behavior is far from optimum, it is, in principle, half of the experiments that result in better control (assuming that the variations are small enough). Second, it is the behavior of the whole model that is being adapted, not only the behavior in the individual state; this means that fewer optimization steps are needed[4]. Faster adaptation could be reached if so called *momentum term,* for example, were included in the updating equation.

In this case, the resolutions of the “critic” are based on the infinite-time quadratic cost criterion — behavior is “good” if this criterion goes down (as calculated over the whole

[4]Of course, one could also store the approximated cost criterion value in the model, so that it could be directly calculated as a function of the state, and based on that, the control could be optimized; however, there are complications, and after all, the random search strategy sounds more “life-like”

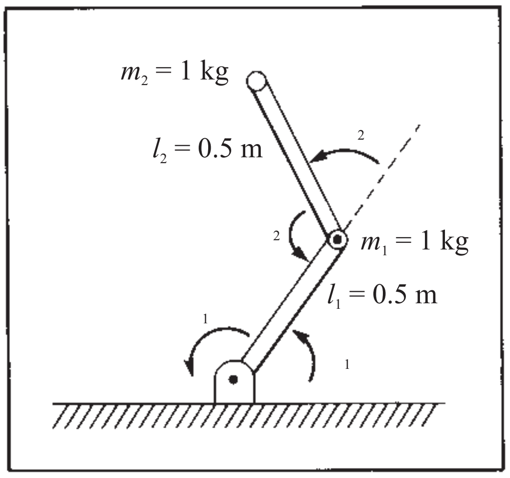

Figure 8: Robot arm being controlled

trajectory):

$$J = \sum_{k=1}^{\infty} (y(k) - y_{\text{goal}})^T Q (y(k) - y_{\text{goal}}) + u^T(k) R u(k). \tag{12}$$

Here, the matrices Q and R are positive (semi)definite weighting matrices that can be changed to alter the resulting closed-loop dynamics. The nice thing about the above criterion is that for affine systems optimizing it results in affine control laws. It is evident that before this criterion can be utilized and before optimization can be started, some kind of stabilizing control that is capable of bringing y to y_{goal} is necessary — otherwise the cost values will be infinite. Additionally, the goal state has to be an equilibrium point, otherwise the contribution of the control never vanishes (u does not go to zero). There are different ways to fix this problem — for example, one can explicitly eliminate the final value of the control:

$$J = \sum_{k=1}^{\infty} (y(k) - y_{\text{goal}})^T Q (y(k) - y_{\text{goal}}) + (u(k) - u_{\text{goal}})^T R (u(k) - u_{\text{goal}}). \tag{13}$$

The output y is used in the criterion because the actual system states ξ cannot be measured (and one does not even know what they are). If some vector elements are not to be weighted, one can put zeros in Q in appropriate locations.

In AI, *reinforcement learning* is an old control strategy that has very much in common with the above critic-based optimization. However, in the AI applications one is typically dealing with very complicated system structures. If the mathematical structure of the model is loose, learning becomes slow and results cannot be quaranteed, and what has already been learned cannot easily and reliably be extrapolated or interpolated beyond the set of actual training samples. This means that in all states, appropriate behavior has to be learned separately! The originally "intelligent-looking" control task has become a problem of storing and managing a body of Chinese Room characters. Now, when looking at the proposed chunk control, one can see that the underlying structure is extremely simple, and powerful models are available. The theoretically sound internal model makes it possible to implement some level of "automated understanding", abstraction, and generalization beyond the training samples can be carried out reliably because of the underlying linear structures.

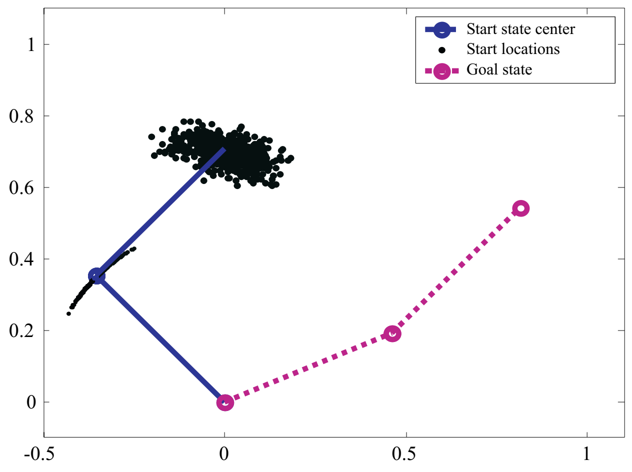

Figure 9: Original and intended final configurations of the robot arm. The actual starting points (locations of the joint and the end-effector) are shown as dots

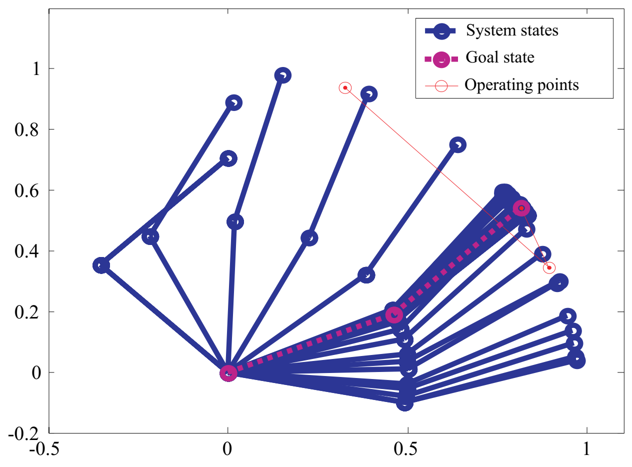

Figure 10: Behavior of the arm when applying the original control law. The "operating points" show the end-effector locations as determined by the C_0 vectors along the three-dimensional SOM after training

5 APPLICATION EXAMPLE

A simple mechanical manipulator was simulated to test the above control scheme. The structure of the robot arm to be controlled is shown in Fig. 8, together with the physical variable symbols and their numeric values. To express the dynamics of the robot arm, one needs a four-state model (for example, a state model could be constructed if the angles θ_1 and θ_2 and their derivatives $d\theta_1/dt$ and $d\theta_2/dt$ were selected as state variables). It is now assumed that all of these state variables can directly be measured (note that this is not necessary, as far as the states are *observable* in the measurements; a set of linearly independent state variables can be extracted by the employed principal component scheme). The number of measurements was now the same as the degrees of freedom in the system, that is, there was now no need of reducing the data space. The control dimension is two: Both of the torques τ_1 and τ_2 can freely be manipulated. The dynamics of the arm is highly nonlinear (see [3]); there also exist singularities in the joint space. However, nonlinearities are smooth and locally linearizable.

When the robot arm dynamics were modeled using the feature model, it was assumed that there was one single movement to be trained. There were two non-optimal SISO PID controllers for controlling both of the torques separately to implement the original stabilizing control (see Figs. 9 and 10; in Fig. 10, a "stroboscope visualization" of the arm behavior is shown). Three linearization centers were employed, each submodel consisting of the linearization center C_0 and four subspace axes ($N = 4$). One of the center vectors

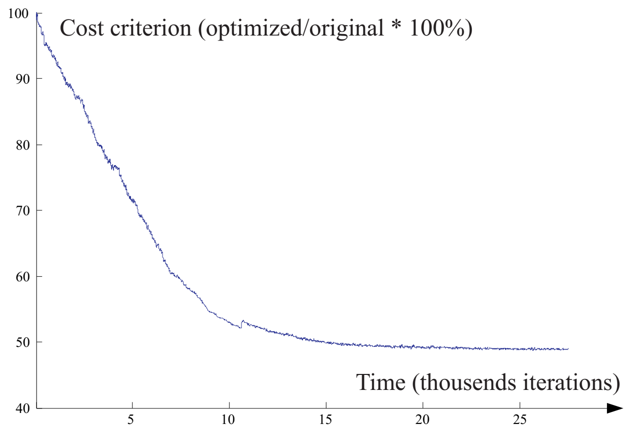

Figure 11: Behavior of the cost criterion during optimization. The process of optimizing the process was by no means optimized!

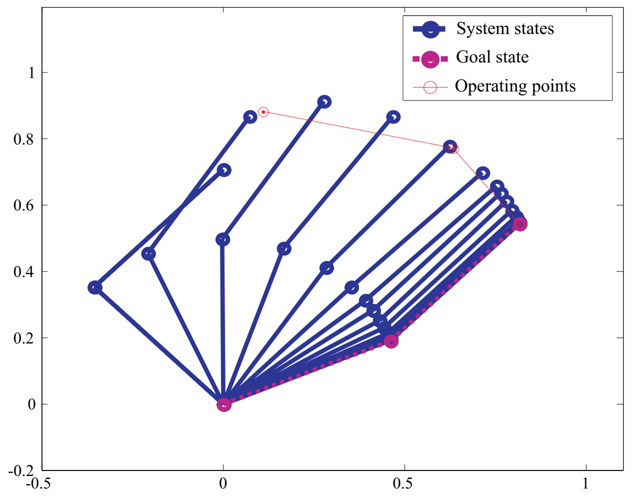

Figure 12: Behavior of the arm when applying the optimized control law

was explicitly fixed to the goal state; this is the simplest way to assure that there never remains steady-state error in the controlled system even though the controls were not yet completely polished. The three submodels were connected by a one-dimensional self-organizing map. In this application, the simple "nearest center" criterion was used, that is, the center point vectors C_0^c were directly compared against y when searching for the best matching model. It turned out that after the training the PID control operation could be restored in an almost errorless manner by the associative control strategy.

When the control was being optimized, all weightings were equal, that is, variations in both angles, their derivatives, and the two torques had the same impact in the optimization criterion (Q and R being unit matrices). The optimization process was slow but the results were convincing (see Figs. 11, 12, and 13). During the second optimization experiment the two angles were weighted ten times more than the other quantities; the control became considerably faster (see Fig. 14).

It needs to be noted that the transfer from a linear model to another may cause discontinuities in the control signal. These transients could be filtered by some additional signal processing elements or applying some "bumpless transfer" schemes; however, as a mechanical system the robot arm is rather robust against transients, being based on a combination of double integrators, so that control signals become automatically smoothened.

Even though the training in the experiment was carried out for the single nominal trajectory, it turned out that the controller (being based on linear structures) can be

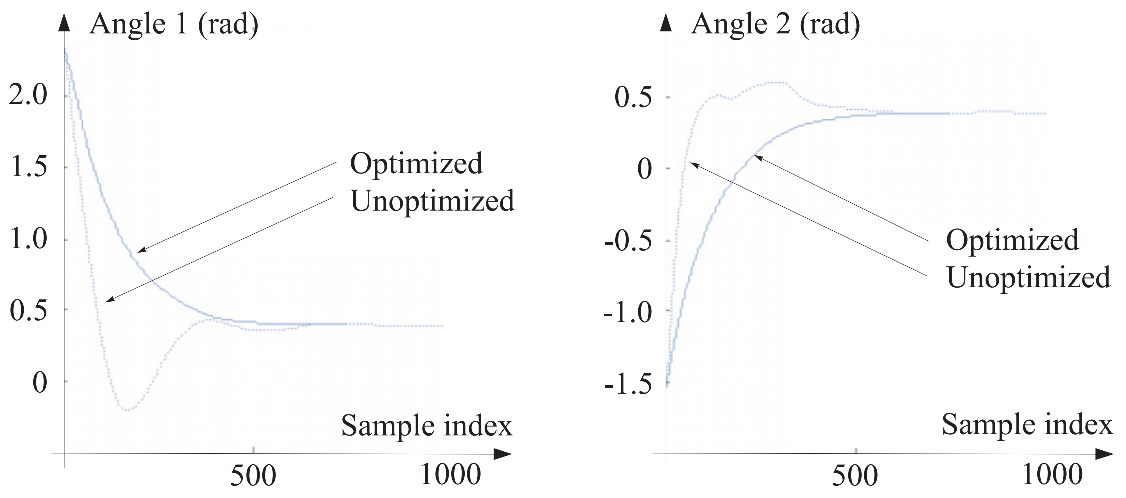

Figure 13: Behavior of the arm angles before and after optimization

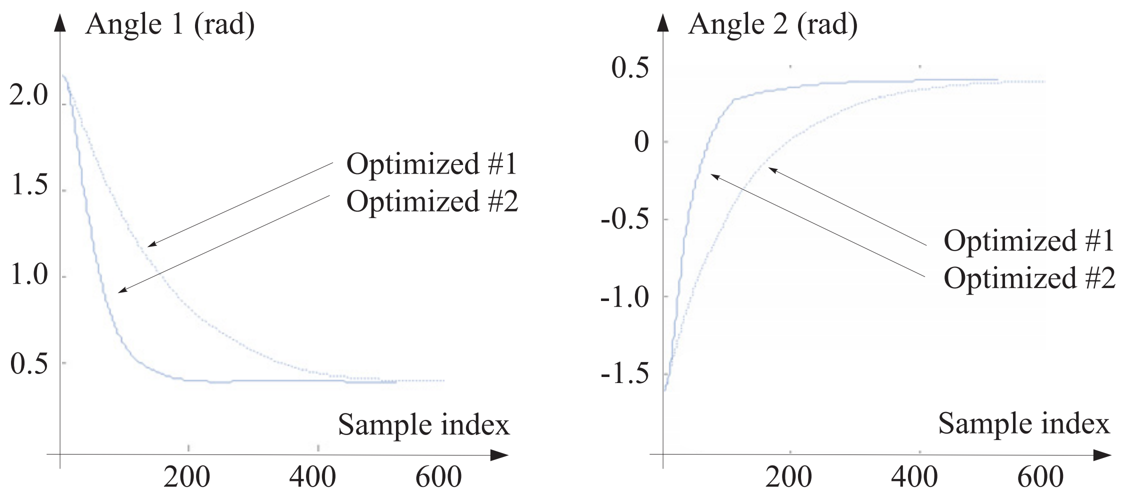

Figure 14: Behavior of the arm angles after "softer" and "harder" optimization

extrapolated and generalized to other trajectories, as long as the goal state remains the same. Even though the control is less optimal, it still works in a rather robust way: It is almost as far as the causalities work in the assumed direction, the control drives the system towards the goal state — and when getting near to that goal state, the available models become more and more appropriate, meaning that the behavior becomes more and more optimal along the trajectory.

As presented in [11], more complicated sequences of movements (like *walking*) can (at least in principle) be modeled and optimized in the same, cognitively motivated numeric framework. It remains to be seen whether such strategies can be implemented in real systems, not only in simulation environments.

ACKNOWLEDGEMENT

I am grateful to Mr. Olli Haavisto who implemented the "cognitive simulation environment" and carried out the actual simulations in `Simulink` during the Summer 2002.

REFERENCES

[1] Åström, K.J.: *Adaptive Control.* Addison–Wesley, 1989.

[2] Åström, K.J. and Wittenmark, B.: *Computer-Controlled Systems — Theory and Design.* Prentice–Hall, Upper Saddle River, New Jersey, 1997 (3rd edition).

[3] Craig, J.J.: *Introduction to Robotics (Mechanics & Control).* Addison-Wesley, 1986.

[4] Haykin, S.: *Neural Networks. A Comprehensive Foundation.* Macmillan College Publishing, New York, 1994.

[5] Hyötyniemi, H.: *Constructing Non-Orthogonal Feature bases.* Proceedings of the International Conference on Neural Networks (ICNN'96), June 3–6, 1996, Washington DC, pp. 1759–1764.

[6] Hyötyniemi, H. and Saariluoma, P.: Chess — Beyond the Rules. In Timo Honkela (ed.): *Games, Computers and People (Pelit, tietokone ja ihminen),* Finnish Artificial Intelligence Society, Helsinki, Finland, 1999, pp. 100-112.

[7] Hyötyniemi, H.: On Mental Images and 'Computational Semantics'. In *Proceedings of the 8th Finnish Artificial Intelligence Conference STeP'98* (eds. Koikkalainen, P. and Puuronen, S.), Finnish Artificial Intelligence Society, Helsinki, Finland, 1998, pp. 199–208.

[8] Hyötyniemi, H.: *Multivariate Regression — Techniques and Tools.* Helsinki University of Technology, Control Engineering Laboratory, Report 125, 2001.

[9] Hyötyniemi, H.: *On Emergent Models and Optimization of Parameters.* 43rd Conference on Simulation and Modeling (SIMS), Oulu, Finland, September 26-27, 2002, Oulu, Finland.

[10] Hyötyniemi, H.: *Studies on Emergence and Cognition — Parts 1 & 2.* Finnish Artificial Intelligence Conference (STeP'02), December 16–17, 2002, Oulu, Finland.

[11] Hyötyniemi, H.: *Towards Perception Hierarchies.* Finnish Artificial Intelligence Conference (STeP'02), December 16–17, 2002, Oulu, Finland.

[12] Jacobs, O.L.R.: *Introduction to Control Theory.* Clarendon Press, Oxford, 1974.

[13] Kohonen, T.: *Self-Organizing Maps.* Springer–Verlag, Heidelberg, 1995.

[14] Longman, R.W.: Iterative learning control and repetitive control for engineering practice. *International Journal of Control,* Vol. 73, No. 10, 2000, pp. 930–954.