Research note

Implementation of an optimal control strategy for a hydraulic hybrid vehicle using CMAC and RBF networks

A. Taghavipour a,∗, M.S. Foumani b, M. Boroushaki c

a Department of Systems Design Engineering, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, N2L 3G1, Canada
b School of Mechanical Engineering, Sharif University of Technology, Tehran, Iran
c Department of Energy Engineering, Sharif University of Technology, Tehran, Iran

Received 2 January 2011; revised 30 October 2011; accepted 25 February 2012

KEYWORDS
Hydraulic hybrid vehicle; Optimal control strategy; Radial Basis Function (RBF) network; Cerebellar Model Articulation Controller (CMAC).

Abstract A control strategy on a hybrid vehicle can be implemented through different methods. In this paper, the Cerebellar Model Articulation Controller (CMAC) and Radial Basis Function (RBF) neural networks were applied to develop an optimal control strategy for a split parallel hydraulic hybrid vehicle. These networks contain a nonlinear mapping, and, also, the fast learning procedure has made them desirable for online control. The RBF network was constructed with the use of the K-mean clustering method, and the CMAC network was investigated for different association factors. Results show that the binary CMAC has better performance over the RBF network. Also, the hybridization of the vehicle results in considerable reduction in fuel consumption.

© 2012 Sharif University of Technology. Production and hosting by Elsevier B.V. All rights reserved.

1. Introduction

The interest in an alternative automotive powertrain has been increasing steadily since problems of energy shortage and air-pollution have become more critical. An alternative for solving this problem is to use Hybrid powertrains. Common types of these kinds of vehicle are equipped with electric elements and are called Hybrid Electric Vehicles (HEVs). But, this electrical equipment is not the only choice, hydraulic equipment is on the other side, and it has some advantages over its electric counterparts, especially for vehicles containing hydraulic equipment on board. For instance, the service life of a hydraulic accumulator as the storage unit is more than a battery, and it has less limitation in absorbing the regenerated braking energy.

In a hybrid vehicle, one of the most important problems is the control strategy, which specifies which component, and for how long, must provide the power needed for the vehicle propulsion. Control strategy design for a hybrid powertrain can be done using different methods. This strategy must be implementable using simple hardware, and, therefore, must be fast enough and memory-efficient. Besides, these considerations should not affect the main mission of the control strategy, which is to reduce fuel consumption.

The objective of the control strategy is to minimize fuel consumption and the emissions of the vehicle, and, therefore, must be optimized to achieve this. A usual control strategy optimization results in a huge database that is usually placed inside a lookup table. But, searching inside a lookup table is a time consuming procedure, and, also, storing the whole database requires considerable memory inside the controller. In order to overcome these problems, a neural network can be trained to replace the mentioned lookup table.

Neural networks have different types, but not all are applicable to this problem. A Cerebellar Model Articulation Controller (CMAC) is a kind of network that has some attractive features, like a fast learning capability and the possibility of efficient digital hardware implementation [1]. Harmon et al. developed an optimal strategy for a small parallel hybrid-electric unmanned aerial vehicle (UAV), using a CMAC controller [2]. Lin and
Chen merged a self-organized CMAC and sliding mode control to make a controller for an inverted double pendulum. They proved that the proposed control system provides a favorable tracking performance for those nonlinear systems [3].

Radial Basis Function (RBF) networks are another kind of neural network used for modeling data in a high dimensional space [4], especially in classification problems. The resulting neurons in the RBF network partitions a multidimensional pattern space into a set of maximum-size hyper-ellipsoid subspaces in terms of the statistical distributions of the training samples [5].

In this study, we compare RBF with the CMAC network for implementing an optimal control strategy on a SPA (Split Parallel Architecture) hydraulic hybrid vehicle.

Zhang et al. [6] applied a rule based control strategy to a parallel hydraulic hybrid heavy vehicle in order to make the engine operate efficiently and recover maximum braking energy.

Liu et al. [7] analyzed the efficiency model of a hybrid system, on the basis of a charging and discharging mode, for a parallel–series hydraulic hybrid vehicle. Based on the optimization they carried out, the simulation results show considerable fuel consumption reduction under the NEDC drive cycle.

Matheson et al. [8] applied a new control strategy, based on power demand, to a hydraulic hybrid heavy commercial vehicle. Results from the research indicate the potential benefits obtained by using Fuzzy Logic as a control tool.

Surampudi et al. [9] discussed various design considerations for the series hybrid hydraulic configuration, based on the fundamental physics of each component. The system-level impact on the sizing of each component, due to considerations such as regenerative braking and performance in duty cycles, was discussed. The control strategy implemented, along with the components picked, indicated a potential fuel economy improvement of up to 20%.

The rest of the paper proceeds as follows. In Section 2, the SPA hybrid powertrain and its control modes are explained. Then, the results of control strategy optimization will be reviewed. Sections 4 and 5 describe the RBF and CMAC networks. Section 6 compares the performance of the two networks. Section 7 compares fuel consumption reduction using the proposed control strategy, with another control strategy on a SPA configuration. Finally, Section 8 provides concluding remarks.

2. SPA hybrid configuration

In SPA architecture, there is a motor just before the final drive, which can assist the engine in propelling the vehicle. This assistance results in a reduction in Internal Combustion Engine (ICE) fuel consumption. A storage unit provides the power for the motor. Also, the motor can be used to capture the energy dissipated, while braking and storing a part of this energy in the storage unit. There is also a generator connected to the ICE that will charge the storage unit. In a hydraulic hybrid powertrain, the motor, the generator and the storage unit are a hydro motor, a pump and a hydraulic accumulator, respectively. Figure 1 shows the schematics of this configuration.

The control strategy for this configuration has 6 control modes, according to Table 1.

Figure 2 shows the proposed controller inputs and outputs for this hybrid configuration, according to the control modes.

In the first mode, the Rear Traction Motor (RTM) will provide the required power. In this mode, the engine is off. This mode is allowable at low speeds when the required driving power is relatively low, and, also, when there is a sufficient State Of Charge (SOC) in the accumulator. As the required driving power increases (usually when vehicle speed is higher), the RTM cannot provide all the power needed. At this stage, the engine with the higher rated power will drive the vehicle and the RTM will be shut off. This is the Engine Only mode. The Split Mode occurs at high speeds and aggressive accelerations. In this situation, the engine alone cannot supply the power needed. So, the RTM can assists the engine and provide the remaining power. The 4th mode is for charging the accumulator. Since the SOC has been decreased during RTM operation in previous modes, the charge must be replaced. The best time to do this is when the rpm of the engine is rather high and the exerted torque is low, for example, when the vehicle is cruising. Here, the hydraulic pump connected to the engine will add some load for charging the empty accumulator, without decreasing the rpm, since the driver will push the accelerator to keep the velocity constant. This mode will make the engine operate at higher torque levels, and this will make the engine status more efficient. The 5th mode is regenerative braking in which the RTM acts inversely, just like a hydraulic pump, and charges the accumulator. After the absolute stoppage of the vehicle and being assured of sufficient SOC in the accumulator, the engine can be shut off to avoid idling. The idling of a vehicle is a period during which the engine operates inefficiently, due to a very low load.

The starting of the engine is the 6th control mode. A constraint on the control strategy is that continuous discharging of the storage unit will never deplete the accumulator, and the charge for starting the engine will remain in the accumulator. Switching between these modes is done by the use of clutches and directional hydraulic valves, which is shown in Figure 1.

3. Optimal control strategy

According to the control modes, one can design a rule based controller for hybrid vehicles. The main problem for designing a control strategy for this vehicle is when organizing the first 3 modes. For other control modes, the strategy has a definite task. For instance, when the driver is braking, the 5th mode...
will be activated by the main controller. A similar action will be taken for the Engine Starting mode. But, for power management in the first 3 modes, the main controller needs a more precise strategy. This strategy depends on the maximum power that can be supplied by the engine and the hydraulic motor. Another important task is identifying the sweet spots on the engine performance map.

In brief, the strategy must be designed in such a way that the engine operates as efficiently as possible. Controlling the engine individually in the SPA configuration is impossible, since the engine is directly connected to the wheels. Therefore, the engine cannot always operate efficiently. Considering the mentioned limitations, an offline global optimization (Dynamic Programming) was developed [10]. The control strategy was optimized according to the Urban Dynamometer Driving Schedule (Figure 3).

The optimization process was done for a SPA powertrain on a van. For this vehicle, the peak powers of the engine and the hydraulic motor are 65 kW and 36.4 kW, respectively.

Dynamic programming works on the basis of a chain of decisions. Therefore, dynamic programming has some inputs at its disposal every moment, and knows what outputs would be achieved in the following moment by introducing each of these inputs [11].

Dynamic programming is executed in three main stages. The first stage is the gridding and division of the input space. A matrix by the name of Cost-to-go is formed at the second stage, and the optimum trajectory is found at the third stage. Out of these three stages, the last one takes a lot of time, because the optimum trajectory is searched from among a large number of points included in the Cost-to-go matrix. Here, there is a state variable that is the same as the accumulator charge. Whereas the energy method is used in this problem, the amount of state variable should transform into energy, too. However, it should be noticed that the relation between SOC and energy is not linear.

Dynamic programming works based on power and energy. In other words, maximum accumulating energy should be achieved. Due to the change of pressure with the volume of fluid inside an accumulator, we cannot say that the energy in mode SOC = 0.5 equals half the maximum energy. The output of dynamic programming shows the situation of the accumulator in terms of energy.

The problem can be formulated according to the following relation (\( \dot{m} \) is the rate of engine fuel consumption):

\[
J(x) = \int_{t_0}^{t_f} \dot{m}(x(k), u_1(k), u_2(k)) \, dt \\
= \int_{t_0}^{t_f} \dot{m}(SOC, \Delta SOC_1, \Delta SOC_2) \, dt.
\]

The constraints on this problem are reviewed as follows:

\[
\begin{align*}
\omega_{ice-min} & \leq \omega_{ice} \leq \omega_{ice-max} , \\
0 & \leq \omega_{rtm} \leq \omega_{rtm-max} , \\
0 & \leq \omega_{gen} \leq \omega_{gen-max} , \\
\omega_{gen} & = \omega_{ice} , \\
0 & \leq T_{ice} \leq T_{ice-max} , \\
T_{rtm-min} & \leq T_{rtm} \leq T_{rtm-max} , \\
0 & \leq T_{gen} \leq T_{gen-max} , \\
0 & \leq P_{ice} \leq P_{ice-max} , \\
\Delta SOC_{min} & \leq \Delta SOC \leq \Delta SOC_{max} , \\
SOC_{min} & \leq SOC \leq SOC_{max}.
\end{align*}
\]

where \( \omega_{ice}, \omega_{rtm} \) and \( \omega_{gen} \) are engine, RTM and generator speeds, respectively. \( T_{ice}, T_{rtm} \) and \( T_{gen} \) are engine, RTM and generator torque, and \( P_{ice} \) is the engine power.

One constraint on the optimization problem is that the states of charge at the first and last time steps are unchanged, in order to guarantee the absence of charge depletion at the end of the simulation [12].

\[
E_{accumulator}(t_f) - E_{accumulator}(t_0) = \int_{t_0}^{t_f} P_{accumulator}(t) \, dt = 0.
\]

\( E_{accumulator} \) is the energy stored in the accumulator. As mentioned, there are two inputs in this problem. One input is the power provided by the hydraulic motor, and the other input is the power received by the pump from the internal combustion engine indirectly. What optimization needs is the minimum and maximum power of these two parts. The optimization process will show the best power split between ICE and RTM, in order to reach minimum fuel consumption. The result of optimization is the power that should be supplied by the hydraulic motor for different driving statuses (Figure 4). It is obvious that the remaining power for vehicle propulsion must be supplied by ICE. The negative values in Figure 4 show the occasions when the driver is braking. So, the hydraulic motor is operating like a hydraulic pump and is charging the accumulator. In order to provide the training data set for the network, the related power request and the accumulator state of charge (SOC) must be known (inputs for the main controller) in addition to the optimized RTM power. The trained network plays the role of a power-based control strategy.

But, the power-based strategies are dependent on the drive cycle. So, there should be another input for the controller that is related to the wheel speed. The requested power (Figure 5),

![Figure 2: The main controller inputs/outputs.](image)

![Figure 3: UDDS drive cycle.](image)
4. RBF neural network

The radial basis function networks employ, as the name suggests, a function with radial behavior around one point (center), and use a distance, usually Euclidean, as the construction method (although other matrices can also be employed). In this network, the nonlinear parameters are the centers \( (c) \) and the spreads \( (\sigma) \), and the nonlinear neurons are grouped into one hidden layer. The main topology of the RBFNN is shown in Figure 8 for 3 hidden nodes. The RBF basis functions, in contrast to the function employed in the MLP (Multi-Layer Perceptron), have local behavior, although not strictly, as it depends on the \( \sigma \) parameter [13]. Besides being universal approximators, RBF networks possess the best approximation property, which does not happen with MLPs [14]. In the incoming input, the output \( y(x) \) is simply a weighted linear summation of the output of the basis functions:

\[
y(x) = \sum_{i=1}^{M} w_i \phi_i(x),
\]

where \( M \) is the number of hidden nodes, \( w_i \) is the corresponding element in the weight vector to a basis function. The basis functions for RBF networks depend on Euclidean distance and \( \sigma \), which is the standard deviation of an input from the center of a cluster. For instance, radial linear, radial cubic, thin plate spline or other functions can be applied as a basis function. The chosen function influences both the modeling and the learning abilities of the network, as well as the learning rule. Among the possible basis functions for RBF networks, Gaussian and inverse multi quadratic functions have localized representation [4]. In the case of the Gaussian basis functions,

\[
\phi_j(x) = \exp \left(-\frac{\|x - c_j\|^2}{2\sigma_j^2}\right),
\]

where \( x \) is the \( n \)-dimensional input vector [4].

Before beginning the learning procedure, a clustering stage must be applied in order to find the cluster center and standard deviation. For clustering, some methods have been proposed, but, here, we apply the \( K \)-mean method [4]. In this method, each input pattern will be collected in a specific cluster, whose center has minimum Euclidean distance from that input. Usually, the number of desired clusters determines the standard deviation and vice versa. But, in the \( K \)-mean method, there is no guarantee to finding any number of clusters for a specific input data. In
such problems, other methods of clustering, or even a modified $K$-mean method, in the literature, might be considered.

There are other methods, like hierarchical clustering. The traditional hierarchical clustering method is an agglomerative approach, which organizes similar branch points into a cluster based on the choice of the distance measure, and, therefore, results in a tree-like dendrogram. This method does not require the number of clusters, $k$, as an input, but needs a termination condition. Usually, this method does not guarantee that the inside similarity of the dendrogram is maximized, because each cluster may consist of several different sub-clusters [15]. Of course, discussing the procedure is not within the scope of this paper.

5. CMAC network

The CMAC ANN was originated by James Albus in 1975. The CMAC is modeled after the method in which the cerebellum learns/stores information and controls reflexive movement, compared to a traditional artificial neural network, which imitates interactions between brain neurons. The CMAC attempts to duplicate the functional properties of the brain instead of its structure [16]. The CMAC can be thought of as an adaptive lookup table. The CMAC is better suited to real-time control, compared to a lookup table, for two reasons: the CMAC can generalize, whereas a lookup table cannot, and the CMAC requires much less memory than a lookup table [4].

The CMAC is a feed-forward, supervised, lattice-based, associative memory network that nonlinearly maps the inputs to a hidden associative memory. The hidden memory is linearly mapped to an adaptive weight vector that generates the output. The output is the sum of the activated weights [2]. The number of training iterations is smaller than that of other neural networks [17].

A typical CMAC neural network structure is shown in Figure 9.

The CMAC network can be considered an associative memory, which performs two subsequent mappings. The first, which is non-linear, projects an input space point into a binary association vector. The association vectors always have $p$ active elements, which mean that $p$ bits of the association vector are ones, and the remaining bits are zeros. $p$ is an important parameter of the CMAC network, and is much less than the length of the association vector, $(p)$ [18,19].

If the CMAC is well defined, a basis function will cover a relatively large area of input space.

The CMAC generalizes, due to the width and overlap of the association cells in the hidden layers [16]. The generalization parameter determines the number of association layers, the number of weights contributing to each output, and the size of support for each basis function [4].

Brassai and Bako studied the hardware implementation of the CMAC type neural network. They found that in the hardware implemented CMAC controller, the following error can be decreased by increasing the number of bits used for parameter representation, and for input coding [20]. Sayil compared different algorithms for training a CMAC network, and recommended an algorithm that has a moderate computation time, and fast initial and long term convergence [21].

6. Control strategy implementation

As mentioned before, we trained an RBF and a CMAC network, according to the results from the previous dynamic programming optimization [4]. The goal function for the learning procedure in both networks is the sum of squares error:

$$E = \frac{1}{2} \sum_{n} (y_{\text{output}} - y_{\text{target}})^2.$$  \hspace{1cm} (15)

The performance of each network will be measured through normalized mean squared error:

$$\text{NRMSE} = \sqrt{\frac{\sum_{n} (y_{\text{output}} - y_{\text{target}})^2}{\sum_{n} (y_{\text{target}})^2}}.$$  \hspace{1cm} (16)

where $Y_{\text{target}}$ is the mean value of all targets.

The database was split into parts randomly. Some data were considered as the training data set, and the remaining data were kept for testing the network performance. The RBF network was constructed for a different number of clusters. But, we found that for the specified training data set, the $K$-mean method could not find more than 8 clusters. The result of training the RBF network is given in Table 2.

The network performance for 2 clusters was too weak to be considered.
It is obvious that it takes longer to place the input data into more clusters, but it can be seen that the network performance does not dramatically change by increasing the number of clusters. Figures 10 and 11 display the training and test data set beside the network target for the best network performance (RBF with 7 clusters).

For constructing a CMAC network for a specific association factor, the number of basis functions or members of the association vector should be determined. According to [4], the number of association vectors depends on the partitioning of the input space. Partitioning the input space must be done in accordance with the data redundancy in different parts of the space.

Where the data density is high, a finer partitioning must be applied, and this is a problem dependent procedure. The number of basis functions is, approximately:

\[
p \approx \prod_{i=1}^{n} \rho_i \quad (17)
\]

where \( n \) is the dimension of the input space, \( \rho \) is the association factor and \( \rho_i \) is the number of intervals in the related input dimension [4].

The network inputs, according to the variation of requested power, engine speed and accumulator state of charge, were divided into 45, 35 and 20 intervals, respectively. Therefore, the number of input cells is \( 45 \times 35 \times 20 = 31500 \).

For instance, if the association factor is 4, there are:

\[
p \approx \frac{45 \times 35 \times 20}{16} \approx 16969. \quad (18)
\]

The next important step is placing the basis functions on the overlays. The number of overlays is equal to the association factor. On each overlay, the input space is divided into some parts, and each part is the domain of a specific basis function. The sum of these domains is equal to the number of basis functions. These overlays are divided using overlay displacement vectors [4].

Here, we used binary CMAC, where the basis function is zero or one. If a pattern is inside the domain of a basis function, the related value of that pattern in the association vector will be one, otherwise, it will be off or zero. So, the association vector is constructed with a great deal of on or off bits. In this way, the CMAC network was constructed and its performance was investigated for different association factors, according to Table 3.

Because of initial partitioning, using an association factor larger than 12 leads to wrong divisions on each overlay. On the other hand, an association factor less than 5 leads to a weak performance of the network. Table 3 shows that the best performance of the CMAC network is obtained for \( \rho = 11 \), and the performance of the network variation with an association factor is more considerable than the RBF network.

### Table 3: CMAC network performance.

<table>
<thead>
<tr>
<th>Association factor</th>
<th>Training set NRMSE</th>
<th>Test set NRMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.6821</td>
<td>0.8773</td>
</tr>
<tr>
<td>6</td>
<td>0.3455</td>
<td>0.3432</td>
</tr>
<tr>
<td>7</td>
<td>0.2072</td>
<td>0.2351</td>
</tr>
<tr>
<td>8</td>
<td>0.2634</td>
<td>0.3558</td>
</tr>
<tr>
<td>9</td>
<td>0.1205</td>
<td>0.1954</td>
</tr>
<tr>
<td>10</td>
<td>0.2304</td>
<td>0.2593</td>
</tr>
<tr>
<td>11</td>
<td>0.0769</td>
<td>0.0597</td>
</tr>
<tr>
<td>12</td>
<td>0.1024</td>
<td>0.1846</td>
</tr>
</tbody>
</table>

Figures 12 and 13 show the training and test data set beside the network target for the best CMAC network performance. The memory occupied for these network is much less than a huge lookup table that contains all possible driving statuses (e.g. for a number of drive cycles). Meanwhile, this network is capable of generalizing the driving conditions, while a lookup table just accesses the limited data stored inside it.

Comparison between Figures 10–13 shows the more desirable performance of CMAC network in following the objective curve. Therefore, the CMAC has a better performance than the RBF network, also, the memory occupation of CMAC is less than that of the RBF network.

In the next section, the resulted network will be tested on a model of the vehicle to highlight the effectiveness of implementing such a control strategy in fuel consumption reduction.

### 7. Effect of implemented control strategy on fuel consumption

In this section, the fuel consumption of the mentioned vehicle is approximated using the developed control strategy. The rate of fuel consumption reduction is compared with the result of another SPA powertrain simulation, which is developed in [22]. The SPA powertrain, which is modeled in [22], is an electric one, and the authors have developed a control strategy based on minimizing powertrain power loss. In that reference, the developed control strategy has reduced fuel consumption by 13.4% or increased fuel economy by 15.5%, according to the UDDS drive cycle.

Using a CMAC neural network, which has been trained based on a dynamic programming solution in this paper, the fuel consumption reduction will be 31.6%.

Figure 14 shows cumulative fuel consumption for the vehicle, before and after hybridization.

The mentioned percent of reduction in fuel consumption will be achieved if the current powertrain of the vehicle is replaced by the SPA hydraulic hybrid powertrain. The developed control strategy is implemented on this powertrain and the ICE is downsized to a peak power of 65 kW, instead of having a rated power of 65 kW. This rate of reduction is considerable, since the electric elements are more efficient than their hydraulic counterparts. Table 4 reviews the achievements of this research.
8. Conclusion

In this study, implementation of an optimal control strategy for a SPA hybrid vehicle was investigated. The controller had 3 inputs. In order to avoid using a lookup table that lacks the generalization capability, neural networks were considered. The behavior of the resulted database motivated us to use networks containing nonlinear mapping. In this way, the capabilities of two types of neural network were compared. RBF and CMAC networks are desirable for this application because of their speed of learning and memory-efficiency. Therefore, they are applicable to online control. For clustering the RBF network input, the $K$-mean method was chosen, and the network performance was compared to the binary CMAC network. This comparison was done for training and test data sets. After constructing the CMAC network for different association factors, and the RBF network for a different number of clusters, the results showed that the binary CMAC (with an association factor equal to 11) has a better performance, and the flexibility of the CMAC network does not decrease for larger association factors.

Finally, the implemented control strategy was simulated on the vehicle model, and it was shown that fuel consumption is reduced by 31.6% when using the SPA hydraulic hybrid powertrain controlled by the developed strategy.

References


Amir Taghavipour received an M.S. degree in Mechanical Engineering from Sharif University of Technology, Tehran, Iran, in 2009. He is now a Ph.D. degree student in the Department of Systems Design Engineering at the University of Waterloo, Canada. His research interests include automotive powertrain control, hybrid vehicles power management strategy design and optimization.

Mahmoud Saadat Foumani received a Ph.D. degree in Mechanical Engineering from Sharif University of Technology (SUT), Tehran, I.R. Iran, in 2002. He was a Faculty member at Semnan University, in Iran, from 2002 to 2006, and is now a Faculty member of the Mechanical Engineering Department at SUT. He teaches undergraduate and graduate courses in the ‘Applied Design Group’, which focuses on mechanical engineering design, vehicle dynamics and chassis design and advanced mathematics.

Mehrdad Boroushaki was born in Iran in 1966. He received a Ph.D. degree in Mechanical-Energy Engineering from Sharif University of Technology, Tehran, Iran, in 2002, where he is currently Associate Professor in the Department of Energy Engineering. His research interests include intelligent systems and control, and identification and optimization of complex nonlinear systems.