Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence

Model Predictive Control with Uncertainty in Human Driven Systems

Alexander Styler
The Robotics Institute
Carnegie Mellon University
astyler@ri.cmu.edu

Illah Nourbakhsh
The Robotics Institute
Carnegie Mellon University
illah@cs.cmu.edu

Abstract

Human driven systems present a unique optimization challenge for robot control. Generally, operators of these systems behave rationally given environmental factors and desired goals. However, information available to subsystem controllers is often incomplete, and the operator becomes more difficult to model without this input information. In this work we present a data-driven, nonparametric model to capture both expectation and uncertainty of the upcoming duty for a subsystem controller. This model is a modified k-nearest neighbor regressor used to generate weighted samples from a distribution of upcoming duty, which are then exploited to generate an optimal control. We test the model on a simulated heterogeneous energy pack manager in an Electric Vehicle operated by a human driver. For this domain, upcoming load on the energy pack strongly affects the optimal use and charging strategy of the pack. Given incomplete information, there is a natural uncertainty in upcoming duty due to traffic, destination, signage, and other factors. We test against a dataset of real driving data gathered from volunteers, and compare the results other models and the optimal upper bound.

Introduction

In this paper we propose a model predictive controller under uncertainty for subsystems in a human driven system. This applies to a special class of systems where operator controls have a significant effect on the inputs to the subsystems, and upcoming inputs have a significant impact on current decisions. Such systems include heterogeneous battery electric vehicles, hybrid electric vehicles, laptops, cell phones, and smart buildings, where smart energy management is highly dependent on operator usage. It is important that the human operator exhibits patterned behavior and generally acts as a rational agent, so predictions can be made at all. Human factors studies by Kleinman and Baron have shown that human operators follow an internal optimal control strategy in response to some inputs and goals (Kleinman and Baron 1970). This suggests that learning a model of a human operator is possible given access to some of the information that they react to. In these listed systems, operator actions heavily impact the duty on the battery or energy source which is managed by a subsystem controller. Despite the natural difficulty of modeling human behavior, work by Pentland and Liu has shown successful prediction of lane changes (Pentland and Liu 1999), and work by Prokop has shown successful trajectory estimation given limited inputs (Prokop 2001). A rule-based environment with limited inputs and outputs facilitates prediction of driver behavior.

Model predictive control (MPC) uses a system model to generate an optimal control sequence for some lookahead in time. When the environment is controlled, or is very predictable, MPC can perform very well given a suitable model. A prediction of duty, or load, is necessary to model how our subsystem will react to various control sequences. A more accurate prediction generally yields better control results, as the effective model for MPC was closer to reality. Our prediction approach uses a data-driven model, or predictor, with no semantic knowledge of the external system nor human operator. This approach has the benefit of being dynamic over time, but it has the downside of requiring good historical data coverage to yield accurate predictions. Work by Aggelogiannaki and Sarimveis, using radial basis function neural nets, has demonstrated data-driven MPC yields good performance in complex chemical systems (Aggelogiannaki and Sarimveis 2008). With enough data, the learned model is a powerful tool for control.

Human operated systems have a natural uncertainty due to incomplete information. While the operator is likely a rational agent, choosing optimal actions for some internal cost function, the information measured and given to the model is incomplete. For example, in a vehicle driving domain, the model might not know the driver's ultimate destination. A fork in the road, where the driver has historically gone either direction, cannot be predicted without this information. Forming an accurate deterministic model is impossible when information is incomplete. If multiple, divergent predictions are likely, using a mean or mode prediction gives insufficient information and can yield poor results. Therefore, it is important to model and exploit this uncertainty when controlling these systems, and we use a modified model predictive control approach to encapsulate uncertainty in the model. Work by Oldewurtel and Parisio has shown that MPC under uncertainty can outperform both rule-based and deterministic controllers for managing smart buildings given an uncertain weather prediction (Oldewurtel and Parisio 2010).

Our model samples from a distribution of predictions, and each sample is weighted in importance by a confidence cal-

Copyright © 2013, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

culation. Then these samples are then given to a controller that generates the best control sequence, scored by a predetermined cost function for some target control goal. Therefore, this single control sequence is generated against the expectation of upcoming duty as represented by these samples. Our approach hopes to capture and exploit both the repeatability and uncertainty in human behavior by using this method. We demonstrate a complete end-to-end control system operating on an electric vehicle simulator using real human driving data.

In the domain of vehicle optimization, previous work by Voort, Dougherty, and Maarseveen has demonstrated significant gains through driver behavior modification (Voort, Dougherty, and Maarseveen 2001). Behavior modification is an important tool for vehicle optimization that has recently made it to production vehicles using innovative display hints. However, machine learning approaches have also demonstrated potential improvements for vehicle energy control without influencing driver behavior. Work by Lin, Peng, and Grizzle has demonstrated the potential for a significantly improved power management controller for hybrid vehicle using Dynamic Programming (Lin, Peng, and Grizzle 2003). Given the future demand, a policy can be trained to optimally control the power source to meet the current demands of the vehicle. This suggests that an approach that can predict the future demand can be useful for optimization. Additionally, work by Moura et al. has demonstrated improved control of hybrids by training a Dynamic Programming policy on prior drive cycles (Moura et al. 2011). Work by Ermon et al. demonstrates a similar result using Dynamic Programming to create a policy using our dataset and simulated vehicle (Ermon et al. 2012). These works demonstrate that historical data can be useful for creating an improved controller for vehicles with heterogeneous energy systems. Through prediction, our approach creates dynamic data selections at run-time, precluding us from using a similar Dynamic Programming approach due to time constraints. However, it allows us to optimize only highly relevant data that is likely similar to the upcoming demand.

This exploitation of behavior patterns suggests possible value for our approach in additional energy management domains beyond our initial exploration. Energy management in mobile phones and portable electronics also exhibit patterned behavior and energy management goals. Component sleeping, screen dimming, and even charging schemes can be improved using additional information supplied by a model prediction. Previous work by Ravi et al. has demonstrated uptime improvement by recommending charging based on learning patterned behavior of the operator (Ravi et al. 2008). Other research by Falaki, Govindan, and Estrin into smart screen dimming has also shown success in controlling component power by modeling small behavior patterns of the operator (Falaki, Govindan, and Estrin 2009). This motivates our generic, semantic-free approach, that can be adapted to any energy management subsystem in human operated devices. Following the generic implementation, additional performance could be realized by then using semantic knowledge in a specific domain.

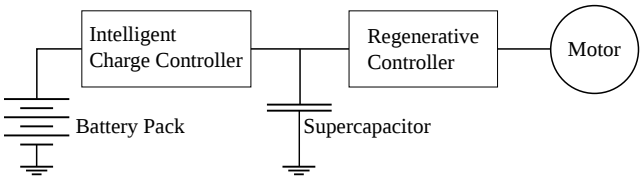

Figure 1: System diagram of heterogeneous energy pack electric vehicle. High power supercapacitor can handle high-energy demands if intelligent charge controller can preemptively charge it from the battery pack.

Heterogeneous Electric Vehicles

While waiting for more sustainable energy options to be realized, it is important to continually improve the efficiency of energy usage in transportation, industry, and homes. Exploiting human behavior patterns can potentially yield savings in all of these sectors, and substantial research has already yielded gains in the work and home. Traffic optimization and electric and hybrid vehicles have already altered the landscape of sustainability in transportation. We explore modeling individual behavior to further optimize the resources at the individual vehicle level.

We examine this approach in a simulated electric vehicle that manages energy in a heterogeneous energy pack. This pack consists of a high power-density, low energy-density supercapacitor paired with a high energy-density lithium iron phosphate battery, as shown in Figure 1. The supercapacitor can handle large currents with great efficiency, but its low energy density makes it ill-suited as a primary energy supply. The lithium iron phosphate battery pack suffers decreased longevity and increased thermal load from high currents, due to lithium fracturing and solid-electrolyte interphase formation. The heterogeneous energy pack provides power from the supercapacitor, if available, during high current events to prevent these loads on the battery.

The driver controls the external system, placing loads on the energy pack and exhibiting repeated driving behavior due to common errands, commutes, traffic patterns, and signage. The internal system provides energy to the traction motor from either the supercapacitor or the battery, and controls the flow of energy between these two different energy stores. It is important to have the supercapacitor charged before high-power acceleration events, and discharged before high-power regenerative braking events. Using MPC the vehicle can preemptively charge the capacitor at a red light, over the course of a minute or more, so that it can handle the acceleration event when the light turns green. This transforms what is normally a brief, high-power load on the battery pack into a long, low-power load on the battery pack.

Dataset and Features

The dataset was collected from seven volunteer drivers who each recorded most of their drives over a one year period. Each driver had a GPS unit, installed in their petrol vehicle, which stored time, elevation, and location each second while driving. We converted this to power versus time for each driver using a standard longitudinal vehicle model of an

electric Honda Civic. Previous work has outlined and validated this model using instrumented electric vehicles (Styler et al. 2011).

The model uses measured and calculated features of the external system to make predictions for duty. From the GPS unit we have: time, elevation, latitude, and longitude. The vehicle model calculates instantaneous velocity, acceleration, and power demand. Additionally we compute various meta-features including: the sum of energy used so far on the trip, the recent variance in the velocity, and the recent variance in acceleration. These features try to capture otherwise unmeasured influences such as road traffic. This set of features makes up the state estimation of the vehicle at each second. The dataset includes the sequences of states matched with the power demand at that state, over all of the trips for each driver for one year. The complete dataset is used as both training and test set using a leave-one-out cross-validation approach. Each trip is tested as if it were the last trip, and all other data is used as prior training examples.

Model Predictive Control with Uncertainty

The nearest neighbor approach is motivated by our test domain, as mapping from the current state of the vehicle to an optimal control is a difficult problem. It is nonparametric, as no polynomial functional regression can successfully map even GPS latitude and longitude to energy control without overfitting problems. Furthermore, precomputing policies is intractable due to the large state space. However, there is a strong relationship between the current state of the vehicle and the upcoming duty. Common routes and traffic patterns dictate repeated duty profiles over the lifetime of a vehicle. Knowing that upcoming duty on a vehicle allows an optimizer to solve for the best sequence of controls to tackle it. Therefore, we use a modified k-nearest neighbor regressor to first map state to upcoming duty, then use an optimizer to solve for the controls.

Algorithm 1 Model Predictive Control with Uncertainty

loop
  $x_t \leftarrow state$
  $\{Z_0, w_0\}...\{Z_k, w_k\} \leftarrow$ PREDICTOR(x_t)
  $\{u_t...u_{t+m}\} \leftarrow$ CONTROLLER($\{Z_0, w_0\}...\{Z_k, w_k\}$)
  EXECUTE(u_t)
end loop

Pseudocode of the algorithm is shown in Algorithm 1. Here, at timestep t, we first measure the state of the vehicle and store it in x_t. This state is then used by the predictor to generate a set of predictions $Z_0...Z_k$ and importance weights $w_0...w_k$.. This set represents k samples from the likely upcoming duty distribution. Finally, a controller or optimizer is used to generate a single control sequence $\{u_t...u_{t+m}\}$ that best handles all samples, up to lookahead m, according to some cost function predetermined for the domain. The first control in the optimal control sequence, u_t is executed, then the entire process is restarted for the next time step.

It is important to note the Markov assumption made in this algorithm. On a given trip at some point in time, the future duty is assumed to be conditionally independent of the past duty given the current vehicle state. Prediction from the past duty to future duty proved very difficult due to delays, phase shifts, and dilation effects from traffic and traffic lights. We roll some information on the past duty into the current vehicle state as the cumulative sum of energy usage, in order to support our Markov assumption.

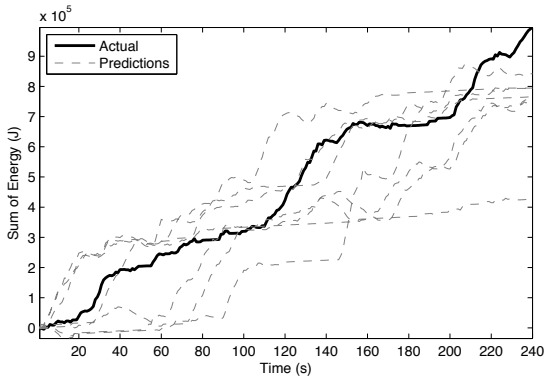

Figure 2: Example set of predictions of upcoming duty versus actual ground truth. Uncertainty can arise from traffic, time delays, or even alternate routes.

The actual implementations of the predictor and controller are discussed in the following sections. The predictor uses a modified k-nearest neighbor regressor, and the controller uses the MPC staple of policy selection optimization. While better results can be expected from more thorough optimizers such as Dynamic Programming, the speed constraint of the domain necessitated a faster approach.

K-Nearest Neighbor Predictor

The predictor uses the current state of the system, represented as a feature vector, and predicts the input to the subsystem. In the case of our test vehicle, the input to the subsystem is the instantaneous power demands each second for some lookahead period. These predictions are used to define the model of the system given to the controller. The cumulative sum of energy over time, with power at each timestep as the slope, is an easier way to visualize these predictions. An example distribution of predictions versus ground truth is shown in Figure 2. The distribution of samples in the example is highly uncertain, but the ground truth is very similar to a few.

Our predictor is a modified version of k-nearest neighbor (KNN) regression. This algorithm compares the distance between the current feature vector and all the prior feature vectors in the past data. It finds the nearest k neighbors by distance and takes the weighted mean of their output to predict the output given the current point. This is known as a lazy-learning approach, where training simply involves storing the data in the past data set, and most calculation is done at search-time. Updating is easy, as points can be added to the set when measured with no additional training required, but the algorithm suffers from potentially long computation at search time and large memory requirements. Search time

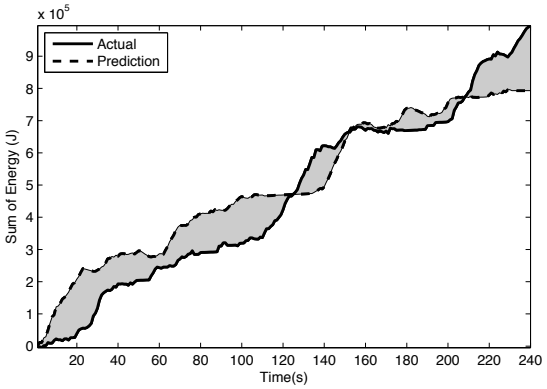

Figure 3: Error calculation for a duty prediction. Shaded area represents error.

is largely mitigated by smart data structures, such as a kd-tree, and memory requirements can be somewhat reduced by data expiration or redundancy removal. Due to our small dataset for our test domain, we have not explored any data expiration techniques to reduce memory requirement.

For our implementation each historical point of driving data maps to the sequence of power usage that followed it. Each of these points is a feature vector, as described above, including latitude, longitude, velocity, etc. The power use versus time is only stored once for each trip, and each point from that trip actually maps to an index into this array. This allows arbitrary lookahead sequences to be returned without having to retrain or reconstruct the tree. Instead of calculating a weighted mean of these sequences, each of these power usage sequences serves as an independent prediction for the controller.

The distance, d, between points is simply a weighted Euclidean distance over the feature vectors,

$$d = \sqrt{\sum_j w_j (x_j - x_j^*)^2} \quad (1)$$

where w_j is a weight for each feature, x is the past point, and x^* is the search point. This distance measures the similarity between states for the current and past points, and the inverse is used to weight the confidence in the match. Weighting each feature manually or creating a custom distance metric both offer a good opportunity for injecting semantic knowledge of the system into the model. Instead, we used a local gradient descent to tune each weight, while keeping the others fixed, and iterating the process until convergence. This generic weighting approach could apply to many domains, but other methods may be necessary to avoid local minima. This weighting demonstrated sufficiently good performance, so a more robust approach to find a global minimum was not pursued. Finally, the k points with the highest confidence are returned to be used by the controller.

While arbitrarily high k can be used, relative confidence, and therefore impact on the output, decreases rapidly with the inverse of distance. High enough k must be chosen to capture the uncertainty, but it must be constrained to keep computation time fast. We use a value of $k = 7$, determined from testing prediction accuracy for different values of k on our validation set. An alternative method, that we did not explore, involves selecting a threshold radius, and using all neighbors within that hypersphere in feature space. This method may yield better results, but can suffer due to uncertain computation time in the controller due to the range of k.

These k neighbors are therefore used as samples from the predicted distribution of upcoming duty. When two samples have similar confidence, but divergent power versus time ground truth, uncertainty is high. If one sample has much higher confidence than others, or all high confidence neighbors have similar ground truth, then uncertainty is low. The variance in the prediction is therefore captured in the weighted samples when passed to the controller. With low variance, the controller can be sure of the prediction and be very aggressive with control. With high variance, however, the controller would yield more a passive control strategy to mitigate cost over the samples. This balance between caution and exploitation is the crux of the gains realized from modeling uncertainty.

We created a heuristic, shown in Figure 3, to calculate prediction error. Given the actual and predicted cumulative sum of energy use, the heuristic represents error as the total area between the functions. This heuristic captures the relative accuracy of a prediction well, despite highly volatile instantaneous power demands and time delays. Having this prediction error allows us modify the prediction step independent of the controller, calibrate feature weighting, and directly observe prediction accuracy.

In Figure 4, we show a histogram of prediction accuracy for all the predictions provided by the model during testing. It is difficult to objective choose a threshold for a good prediction, but the distribution shape lets us compare the median and worst performances. We observe a large volume of predictions falling around the median, with significantly less error than the tail of the distribution. While this is useful to analyze the coverage of the data set, this heuristic cannot be used to compare the accuracy of a set of samples with the accuracy of its mean. This prevents us from comparing the sample-based stochastic KNN approach with standard KNN regression using this heuristic. To compare these, we will have to introduce a controller and compare final performance.

Optimization and Control

The next stage of our system is the model predictive controller that translates these weighted samples of the upcoming duty into a cost-minimizing control for the next timestep. The system then executes this control, measures the new state, makes new predictions, and calls the controller again for the next step. While the controller will generate a sequence of controls each timestep to cover the entire lookahead period, the controls are discarded after the first in the sequence is executed. This follows our Markov assumption, where the next state and its predictions should supersede those of the past state.

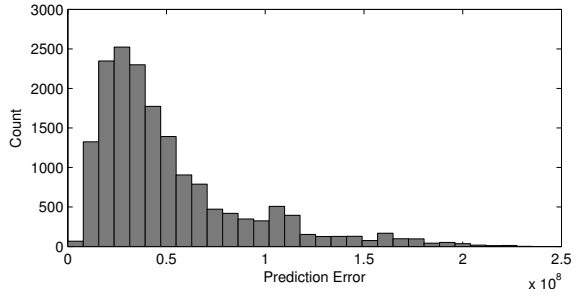

Figure 4: Histogram of prediction accuracies over test data, showing the general shape of the distribution.

The controller minimizes a cost function

$$C(x_t...x_{t+m}, u_t...u_{t+m}, z_t...z_{t+m}) = c(x_t, u_t, z_t) + \gamma * C(x_{t+1}...x_{t+m}, u_{t+1}...u_{t+m}, z_{t+1}...z_{t+m}) \quad (2)$$

where $\{x_t...x_{t+m}\}$ is a sequence of states, $\{z_t...z_{t+m}\}$ is a sequence of inputs given by a prediction, $\{u_t...u_{t+m}\}$ is a sequence of controls, $c(x_t, u_t, z_t)$ is the immediate cost, and γ is a discount factor. Given the inputs, controls, and initial state of the subsystem, we can calculate the sequence of states of the subsystem using a plant, $x_{t+1} \leftarrow H(x_t, u_t, z_t)$. In the test domain, the immediate cost is calculated as the resulting current on the battery squared for that timestep. The charge of the capacitor is tracked in x_t, and if power demand, z_t, exceeds the ability of the capacitor, the net power is supplied from the battery. Squaring the current on the battery helps to penalize higher loads more steeply, resulting in a behavior that favors long low-power drains over quick high-power drains.

Given a plant of the subsystem, a sequence of inputs given by the predictions, and a cost function, most traditional optimizers will work to generate the control sequence. An important caveat is that it must test and score a control sequence, $U = u_t...u_{t+m}$, against a set of predictions, $Z_0...Z_k$, where $Z_i = z_t...z_{t+m}$. The controller therefore chooses the control that minimizes the expected cost over the distribution of predictions, according to

$$U^* = \arg\min_U \sum_{i=0...k} w_i * C(X, U, Z_i) \quad (3)$$

where U^* is the optimal sequence of controls, and w_i is the importance weight of each sample from the prediction model.

The ideal approach for many domains would be to do optimal control at this stage. However, for our domain this is not practical as computation times on the vehicle must be quick to allow for replanning every second. As we have a 240 second lookahead, 7 different predictions, and a continuous charge state, computing an optimal control would require too much computation time for our time constraint. To allow for quick execution on a low power system, a cruder approach is necessary. The controller, however, is interchangeable based on domain needs.

For our controller, a fixed set of 12 possible control sequences is considered. These sequences are all constant controls over the period, chosen log-uniformally over the space of controls. The controller tests each possible control sequence against the predictions, and returns the best scoring result. Even if the control sequences are constant, a complex control strategy can emerge due to the stochastic switching each second from replanning. In validation testing, these control sequences yield good coverage and performance. Additional control sequences add marginal benefit at increased computation cost, and are not necessary in practice.

While we only demonstrate single-variate optimization, the controller is agnostic to the cost function. A more complex cost function or even multivariate optimization can be used in our approach.

Table 1: Reduction of Current-Squared for Each Driver

	Model		
Driver	Mean KNN	Distributed KNN	Prescient
Driver A	53.5%	57.8%	67.5%
Driver B	40.5%	43.6%	49.1%
Driver C	56.2%	61.7%	69.0%
Driver D	37.4%	39.4%	44.9%
Driver E	46.8%	47.1%	57.3%
Driver F	38.4%	38.2%	47.9%
Driver G	59.0%	65.8%	70.9%
Total	**47.8%**	**51.9%**	**55.9%**

Results

We evaluate the success of model predictive control under uncertainty using our modified KNN model. This algorithm was run on our test domain to control the management of energy in an electric vehicle using a heterogeneous battery pack. Success was measured by cost function total: the reduction in current-squared on the battery pack for our simulated heterogeneous energy pack electric vehicle.

Using the selection controller described in the previous section, we test performance of three different models for the algorithm. The first is a standard KNN regression model that returns a mean prediction. The second model is the modified KNN model proposed in this paper that returns samples of the distribution of predictions. The final model is a prescient model that returns the actual upcoming duty with absolute certainty. This model yields an effective upper bound on performance for this controller. Total results for each driver are displayed in Table 1. Results show the reduction in current-squared on the battery pack given prediction from each model, tested over our whole data set using a leave-one-out cross validation approach.

The algorithm has varied success on different trips, due to varied coverage of the space in the past history set and outliers. On frequent trips, such as daily commutes, coverage is very good and better predictions are obtained. In new or unique trips, however, prediction can only match other features instead of GPS, resulting in poorer prediction and performance. On unique routes, upcoming hills and road

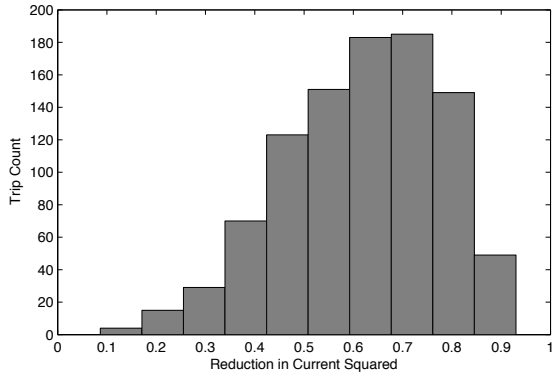

Figure 5: Histogram of performance for all trips. A high variance in performance is observed, with median around 70%

system information is not captured by the data, but other similarities can still be matched. Therefore the prediction yields some information, but not enough to realize significant improvements for these trips. A distribution over the reductions in current squared for all tested trips is shown in Figure 5. We observe here that the median trip experiences around a 70% reduction, and the majority of trips fall around the mean.

The results show a marked improvement towards optimal versus an expected value approach. Over half of the disparity between MPC performance using a mean prediction and the prescient upper bound is reduced by our stochastic approach, improving from a 47.8% reduction to a 51.9% reduction. The approach yields gains for all but one operator, whose driving data consisted largely of static highway commutes. These results suggest that our stochastic k-nearest neighbor model is effectively capturing some uncertainty that can be exploited the controller. This allows the controller to behave more intelligently during high-uncertainty events, balancing the costs of the worst case scenario with the gains of the best case. While the absolute difference in performance between the methods is low, other domains with increased uncertainty may yield larger gains.

However, while average performance is improved, there are no guarantees that can be made. MPC will fail when using an incorrect model, and past data may inaccurately represent the distribution of upcoming duty. In domains where safety or failure is critical, optimizations may devolve into always mitigating a worst-case scenario, precluding the ability to optimize at all.

Future Work

There are many areas of work that can be explored to expand or improve this approach. As the dataset is continually appended with new data, some sort of data expiration method would need to be implemented to maintain guarantees on computation time. Data that is rarely used, frequently incorrect, or simply old could potentially be prioritized for this expiration. Additionally, some sort of automatic redundancy detection or coverage calculation could also be used to combat this issue. Also, clustering methods could be explored to give additional information for the confidence calculation and prediction. Given more computation time for certain domains, more thorough controllers can be tested, or controllers closer to optimal using some intelligent precomputation. Precomputing value graphs or policies for every historical trip could allow more optimal control in the real-time constraints. The prediction accuracy and performance could be further improved in implementation by exploiting semantic domain knowledge in feature distance or weight calculations. Here we have simply demonstrated a generic method for modeling human operator uncertainty, a means to exploit it in subsystem optimization, and example performance on our test domain.

Conclusions

In this paper we presented a model predictive controller (MPC) for optimizing subsystem control under uncertainty in a human driven system. The system is ultimately invisible to the primary system operator and does not directly influence behavior. Using a modified k-nearest neighbor regressor, we generate samples of upcoming duty by feature matching our current state to historical states in our data set. The samples are passed to a controller that chooses a control sequence to minimize a cost function over the lookahead of the predictions. The approach was designed to be data-driven, not exploiting any semantic knowledge about the system in the prediction or controller. The approach is also dynamic, allowing for simple retraining as more data is appended to the data set.

The algorithm shows improved performance versus an expected value algorithm for our chosen controller, and using past data as samples for future duty displays excellent performance for our chosen domain. Our approach may generalize to other energy management domains where the overall system is similarly operated independently of the subsystem goals. This could include energy and component management in portable electronics, hybrid vehicle power selection, electric vehicle thermal control, or even smart building management. As energy usage and efficiency become more scrutinized, intelligent methods for efficiency optimization must be explored.

Acknowledgements

The authors gratefully acknowledge the contributions of Bosch, Bombardier, Google, the ChargeCar team, the CREATE Lab, our data volunteers, and reviewers. This material is based upon work supported by the National Science Foundation under Grant No. (0946825).

References

Aggelogiannaki, E., and Sarimveis, H. 2008. Nonlinear model predictive control for distributed parameter systems using data driven artificial neural network models. *Computers & Chemical Engineering*.

Ermon, S.; Xue, Y.; Gomes, C.; and Selman, B. 2012. Learning Policies for Battery Usage Optimization in Electric

Vehicles. In *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg.

Falaki, H.; Govindan, R.; and Estrin, D. 2009. Smart screen management on mobile phones. *Energy*.

Kleinman, D., and Baron, S. 1970. An optimal control model of human response part I: Theory and validation. *Automatica*.

Lin, C.-C.; Peng, H.; and Grizzle, J. 2003. Power management strategy for a parallel hybrid electric truck. *IEEE Transactions on Control Systems Technology*.

Moura, S. J.; Fathy, H. K.; Callaway, D. S.; and Stein, J. L. 2011. A stochastic optimal control approach for power management in plug-in hybrid electric vehicles. In *IEEE Transactions on Control Systems Technology*.

Oldewurtel, F., and Parisio, A. 2010. Energy efficient building climate control using stochastic model predictive control and weather predictions. *American Control Conference*.

Pentland, A., and Liu, A. 1999. Modeling and prediction of human behavior. *Neural Computation*.

Prokop, G. 2001. Modeling human vehicle driving by model predictive online optimization. *Vehicle System Dynamics*.

Ravi, N.; Scott, J.; Han, L.; and Iftode, L. 2008. Context-aware battery management for mobile phones. In *Pervasive Computing and Communications*.

Styler, A.; Podnar, G.; Dille, P.; Duescher, M.; Bartley, C.; and Nourbakhsh, I. 2011. Active management of a heterogeneous energy store for electric vehicles. In *Integrated and Sustainable Transportation Systems*.

Voort, M. v. d.; Dougherty, M. S.; and Maarseveen, M. v. 2001. A prototype fuel-efficiency support tool. *Transportation Research*.