Model Predictive Control of Residential Baseboard Heaters with Distributed System Architecture

Eric M. Burger, Hector E. Perez, Scott J. Moura

Abstract— **Typical residential HVAC systems employ mechanical or hard-coded deadband control behaviors that are unresponsive to changing energy costs and weather conditions. In this paper, we investigate the potential of electric baseboard heaters to maintain a comfortable temperature while optimizing electricity consumption given weather forecasts and price data. We first propose a distributed system architecture that utilizes mobile application platforms. We then develop, assemble, and deploy a sensor network and Internet server to collect real-time temperature data from an apartment. With these sensor streams, we identify a thermal model of the apartment. Finally, we propose a model predictive control algorithm and perform a software-in-the-loop simulation of the cloud-based system to demonstrate the economic advantage.**

I. INTRODUCTION

Commercial and residential buildings account for 41% of U.S. primary energy consumption, more than either the transportation sector or the industrial sector (29% and 30%, respectively) [1]. Heating, ventilation, and air-conditioning (HVAC) compose 43% of commercial and 54% of residential building site energy end-use. Space heating alone accounts for 45% of residential site energy use and is the focus of this work.

A common system for residential space heating is electric resistance baseboard heaters. Such systems are cheap to produce but costly to operate due to poor primary to end-use energy efficiency ($\sim$30%) and the varying cost of electricity [2]. However, as long as comfort levels are maintained, homeowners are indifferent to precisely *when* energy is used. Additionally, because homeowners often pay time-of-use electricity rates, we can know the price of electricity *a priori* and use the price as a proxy for power grid demand. By forecasting the space heating demand and incorporating the electricity price schedule, residential space heating can be controlled to minimize electricity costs. In practice, this optimal control strategy performs load shifting to avoid high electricity prices, thereby reducing peak loads on the grid.

The potential advantages of such load shifting, as well as models of thermal zones, system identification techniques, and model predictive control (MPC) strategies, are well developed in the literature. For example, [3] details the modelling, parameter estimation, and validation of a variable air volume (VAV) HVAC system. The work in [4] and [5] propose frameworks for online estimation of states and unknown parameters of buildings using extended and unscented Kalman filters. A model predictive controller is implemented in [6], [7], and [8] which optimizes the HVAC system to minimize total energy consumption, peak power consumption, and/or total comfort violations.

In these and other related works, the system architecture employed to implement these components is either briefly mentioned in the results, proposed in the conclusion, or entirely left out. To address this limitation, we have developed a distributed cloud-based system architecture for a residential heating system using readily available electronics and popular application development platforms. While we have selected a basic thermal model and a single system ID and MPC algorithm, future work will focus on comparing various alternatives to determine their relative advantages.

In this work, we demonstrate how these pieces can be linked into a robust and adaptable system. Our cloud-based system architecture enables us to remotely monitor and modify the control algorithm. Using mobile application platforms, we are able to move computationally intensive tasks to the cloud, thereby reducing the cost and complexity of local hardware. By breaking up the system's resources (database, linear program solver, model predictive controller, etc.) into distinct components, it becomes possible to modify sections of the system without reconstructing the whole. These system characteristics enable the implementation and continued development of the thermostatic control strategies presented in literature.

In this paper, we propose a distributed cloud-based system architecture for a residential heating system. We then deploy a sensor network, local computer, and Internet server to collect real-time temperature data from an apartment. Using the sensor data, we perform system identification on the apartment using the online gradient update law algorithm. Next, we propose a model predictive control algorithm to minimize the cost of the heating system. Finally, we test the cloud-based system with simulated sensory input to demonstrate the economic advantage of the control strategy.

II. SYSTEM ARCHITECTURE

A fundamental challenge of a "smart" or "predictive" control system is access to the information and computing power needed to forecast and respond to future conditions. Therefore, we begin by proposing a distributed cloud-based system architecture. The term "distributed" refers to the incorporation of several autonomous computers that communicate with each other by passing messages. A distributed network can consist of different types of computers, each

E. M. Burger, H. E. Perez, and S. J. Moura are with the Energy, Controls, and Applications Lab (eCAL), 611 Davis Hall, Civil and Environmental Engineering Department, University of California Berkeley, Berkeley, CA, USA. E-mail: ericburger@berkeley.edu

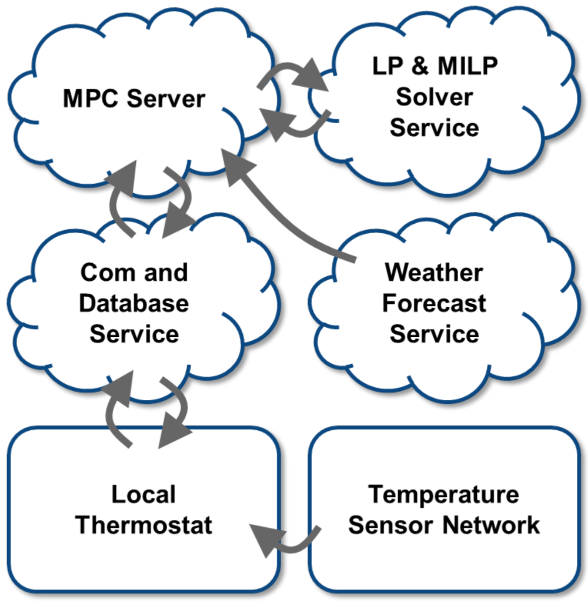

Fig. 1: System Diagram

with an incomplete view of the system. The term "cloud-based" refers to the use of Internet-connected servers which allow a client computer to connect and perform a task from anywhere in the world.

The system architecture implemented in this work is shown in Fig. 1 where the rectangles represent physical hardware installed in the apartment, the clouds represent Internet servers or services, and the arrows indicate the flow of information. The temperature sensor network consists of six nodes placed throughout the apartment and communicating wirelessly with the ZigBee protocol [9]. Each sensor node, shown in Figure 2, includes a Honeywell HIH6130 sensor, a Digi XBee S2 radio, and an Arduino Pro Mini board with an ATmega328 microcontroller. The local thermostat is an Internet-connected BeagleBone Black microcomputer that acts as a gateway between the heating system, the sensor network, and the cloud-based system.

The system parameters, electricity price schedule, optimization program, and control algorithm are coded into the MPC server and hosted by the mobile-application platform, Google App Engine [10]. Each of the cloud-based services include an application programming interface (API) built on the hypertext transfer protocol (HTTP) with JavaScript Object Notation (JSON) response format. This enables reliable communication across computing platforms and languages, an essential characteristic for our distributed system. The communication and database service, provided by Xively, implements an API to send, store, and retrieve time-series data [11]. This enables the thermostat and MPC server to asynchronously pass messages. The linear program (LP) and mixed integer linear program (MILP) solver is a service developed as part of this work. It consists of a solver hosted on the mobile-application platform, Heroku, and accompanied by an API for sending linear programs in a matrix/vector format [12]. Finally, the weather forecast is provided by the

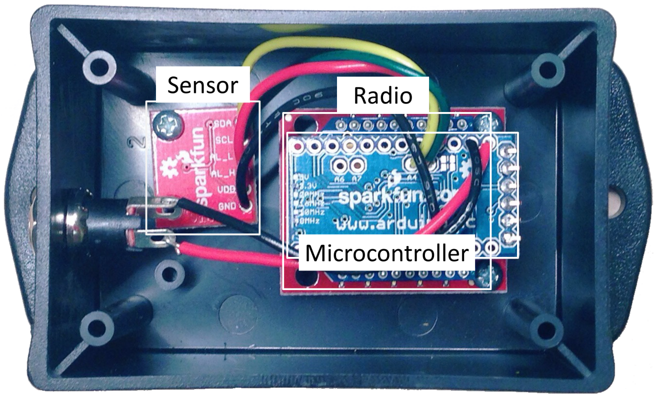

Fig. 2: Sensor Node

Internet service, Weather Underground [13].

The system is executed as follows

- Sensor Network: Once per minute, the sensor network collects temperature measurements and sends them to the thermostat, which stores them in the database service. In this work, only one of the temperature streams is used for system ID and control.
- Weather Forecast: Once every 15 minutes, The MPC server retrieves and parses the weather conditions and temperature forecast.
- MILP: Once the forecast is parsed, the MPC server constructs the matrices of the MILP with the current sensor reading, weather forecast, and electricity price schedule for a 6 hour time horizon (N=24). These matrices are sent to the LP solver service.
- Optimal Solution: Once the problem is solved by the LP solver service, the response is parsed by the MPC server and the optimal heater state (ON or OFF) is sent to the online database service.
- Thermostat: Once every 5 minutes, the local thermostat checks the database service for a change in the optimal heater state record.

A necessity of any distributed computing architecture is fault tolerance. In this case, the local thermostat is programmed to fall back to simple deadband control in the case of communication failure or an infeasible solution from the LP solver service. The deadband control is implemented such that when a lower bound temperature is reached, the apartment is heated for one time step. Because the MPC server is only capable of load shifting (instructing the thermostat to turn the heater on before the lower bound temperature is reached), this fall back does not conflict with our control strategy.

Overall, this work is intended to serve as a demonstration of how distributed and cloud-based systems can aid engineers in developing smarter control systems. By utilizing platforms for electronics hardware, mobile-applications, and Internet services, it is possible to quickly prototype a distributed control system.

III. MODEL DEVELOPMENT & ID

A. Thermal RC Model

This work employs a simple continuous time RC model to capture the thermal characteristics of the apartment. The modeled dynamics include the heat transfer between the interior and the environment, power from the electric baseboard heaters, and solar heat gain. Radiative heat transfer from the ambient will not be considered. Therefore, the change in temperature within the apartment can be represented by the state equation

$$\dot{T}(t) = \frac{-1}{RC}T(t) + \frac{1}{RC}T_{\infty}(t) + \frac{P_H}{C}h(t) + \frac{1}{C}P_S(t), \quad (1)$$

where $T(t)$, $h(t)$, $T_{\infty}(t)$, $P_S(t)$ are the indoor temperature (state, $^{\circ}$C), heater state (input, binary), outdoor temperature (disturbance input, $^{\circ}$C), and solar gain (disturbance input, kW), respectively. The parameters used in this model are R ($^{\circ}$C$/kW$), C ($kJ/^{\circ}$C), P_H (kW) which represent the thermal resistance, thermal capacitance, and baseboard heater power, respectively. In this paper, we will use "solar gain" to refer to power delivered to the apartment attributable to solar irradiance (kW/m^2).

B. Exogenous Signals

To utilize (1) in a model predictive controller, it is necessary to forecast the disturbance inputs, $T_{\infty}(t)$ and $P_S(t)$. Fortunately, there are several web services that provide searchable application programming interfaces (APIs) for temperature forecasts. Such services include Wundergound.com, OpenWeatherMap.com, and Forecast.io. Unfortunately, at the time of this writing, there is no equivalent service for solar irradiance forecasts, making it difficult to accurately predict solar gain. Therefore, we have created a means of approximating solar gain based on forecasted weather conditions. First, we redefine solar gain

$$P_S(t) = P_{S,max}s(t), \quad (2)$$

where $P_{S,max}$ is the maximum solar gain (kW) and $s(t)$ is a solar gain scaling factor. In this paper, we assume that $P_{S,max}$ is constant, however $P_{S,max}$ will actually vary according to the position of the sun (i.e. season and time of day). Next, we use forecasted cloud cover conditions to define the scaling factor, $s(t)$, shown in Table I. Between sunrise and sunset, the input $s(t)$ is equal to a normalized number based on the weather condition. Otherwise (i.e. at night), $s(t) = 0$. This approach provides a crude but useful approximation of the impact of solar irradiance on the temperature inside the apartment.

Finally, (1) can be rewritten as

$$\dot{T}(t) = \frac{-T(t) + T_{\infty}(t)}{RC} + \frac{P_H h(t)}{C} + \frac{P_{S,max}s(t)}{C}. \quad (3)$$

TABLE I: Solar Gain Scale

Weather Condition		Scale	Normalized, s(t)
Day	Clear	4.5	1
	Scattered Clouds	3.5	0.778
	Partly Cloudy	3	0.667
	Mostly Cloudy	2	0.444
	Overcast	0.5	0.111
Night	N/A	0	0

C. Parametric Modeling

We reformulate the temperature dynamics equation into a linear in the parameters form for identification purposes. The linear model is derived from (3) as follows,

$$\dot{T}(t) = \begin{bmatrix} \frac{1}{RC} & \frac{P_H}{C} & \frac{P_{S,max}}{C} \end{bmatrix} \begin{bmatrix} T_{\infty}(t) - T(t) \\ h(t) \\ s(t) \end{bmatrix}. \quad (4)$$

Let

$$z(t) = \dot{T}(t), \quad (5)$$

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{RC} \\ \frac{P_H}{C} \\ \frac{P_{S,max}}{C} \end{bmatrix}, \quad (6)$$

$$\phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \begin{bmatrix} T_{\infty}(t) - T(t) \\ h(t) \\ s(t) \end{bmatrix}. \quad (7)$$

Therefore,

$$z(t) = \theta^T \phi. \quad (8)$$

D. Identification Algorithm

For parameter estimation, we employ a normalized recursive gradient update law, given by,

$$\frac{d}{dt}\hat{\theta}(t) = \Gamma\epsilon(t)\phi(t), \qquad \hat{\theta}(0) = \hat{\theta}_0, \quad (9)$$

$$\epsilon(t) = \frac{z(t) - \hat{\theta}(t)\phi(t)}{m^2(t)}, \quad (10)$$

$$m^2(t) = 1 + \hat{\phi}(t)\phi(t), \quad (11)$$

E. System Identification Results and Validation

With $\Gamma = 10^{-4.1} * I$ and initial parameter guesses $\theta_1(0) = 0.00024$, $\theta_2(0) = 0.00045$, and $\theta_3(0) = 0.000145$, the resulting parameter estimates are shown in Fig. 3. The dataset used for identification is shown in Fig. 4, where $\hat{T}(t)$ is the estimate and $T(t)$ is the measurement. Due to the lack of persistent excitation in regressor element $h(t)$, θ_2 required manual tuning to find a suitable estimate [14]. In other words, the heater was not turned on often enough for θ_2 to be identifiable using the recursive gradient descent algorithm.

For model validation, we rewrite the system in state-space form,

$$\dot{T}(t) = \begin{bmatrix} -\theta_1 \end{bmatrix} \begin{bmatrix} T(t) \end{bmatrix} + \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \end{bmatrix} \begin{bmatrix} T_{\infty}(t) \\ h(t) \\ s(t) \end{bmatrix}, \quad (12)$$

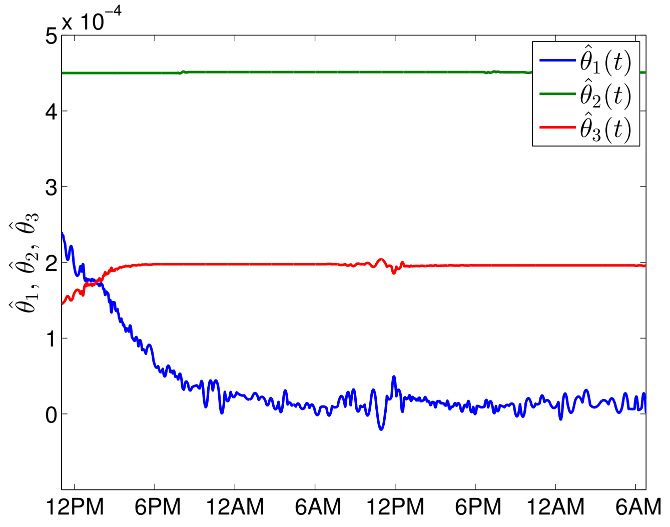

Fig. 3: Parameter Estimates

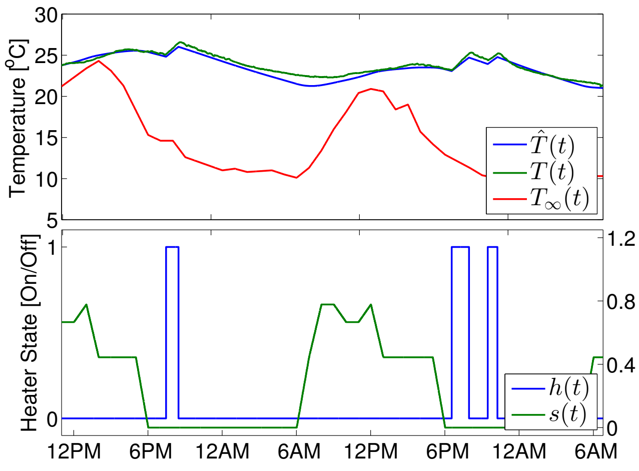

Fig. 4: System Identification Dataset

$$u(t) = \begin{bmatrix} T_\infty(t) \\ h(t) \\ s(t) \end{bmatrix}. \tag{13}$$

From Fig. 3, the exit estimates are

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} 1.054 * 10^{-5} \\ 4.51 * 10^{-4} \\ 1.960 * 10^{-4} \end{bmatrix}. \tag{14}$$

Using these parameter estimates and the state-space model, we can validate against a different dataset, shown in Fig. 5. The validation results show a root mean square error of 0.9130 °C.

IV. MODEL PREDICTIVE CONTROL

A. Model Discretization

Since the model we identified above is in the continuous time domain, we now transform the model into the discrete time domain for use in the optimization program.

Recall the continuous time state-space model, (12), where

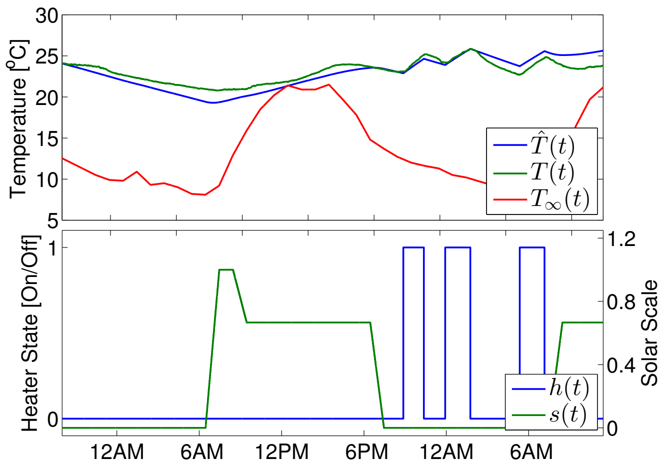

Fig. 5: Validation Dataset

$$A = \begin{bmatrix} -1.054 * 10^{-5} \end{bmatrix}, \tag{15}$$
$$B = \begin{bmatrix} 1.054 * 10^{-5} & 4.51 * 10^{-4} & 1.960 * 10^{-4} \end{bmatrix}. \tag{16}$$

We obtain the discrete time state-space form

$$T(k+1) = A_d \begin{bmatrix} T(k) \end{bmatrix} + B_d \begin{bmatrix} T_\infty(k) \\ h(k) \\ s(k) \end{bmatrix}. \tag{17}$$

by using the transformations

$$A_d = e^{A\Delta t}, \tag{18}$$
$$B_d = \left(\int_0^{\Delta t} e^{A\tau} d\tau \right) B. \tag{19}$$

Since the A matrix is non-singular, we can find B_d as

$$B_d = A^{-1}(A_d - I)B. \tag{20}$$

We solve for these discrete matrices with Δt = 15 minutes = 900 seconds, resulting in

$$A_d = \begin{bmatrix} 0.9906 \end{bmatrix}, \tag{21}$$
$$B_d = \begin{bmatrix} 0.009441 & 0.4040 & 0.17557 \end{bmatrix} \tag{22}$$

B. Optimization Formulation

The optimization program is formulated as

$$J = \min_{h(0)} \sum_{k=1}^{N-1} c(k) P_H h(k) \frac{\Delta t}{3600}, \tag{23}$$

subject to

$$T(k+1) = A_d \begin{bmatrix} T(k) \end{bmatrix} + B_d \begin{bmatrix} T_\infty(k) \\ h(k) \\ s(k) \end{bmatrix}, \quad k = 0, ..., N-1, \tag{24}$$

$$T(k+1) \geq T_{min}, \qquad k = 0, ..., N-1, \tag{25}$$
$$h(k) = \{0, 1\}, \quad k = 0, ..., N-1, \tag{26}$$
$$T(0) = T_0. \tag{27}$$

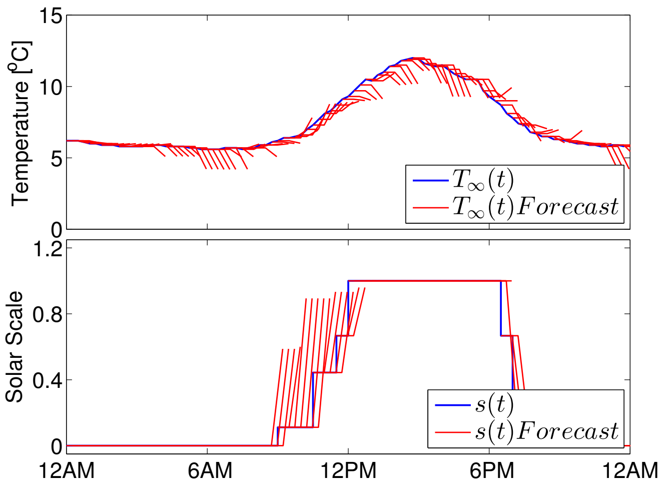

Fig. 6: Weather Forecast Data

The optimization program is a mixed integer linear program (MILP) which minimizes the cost of electricity while keeping the indoor temperature above a minimum setpoint. Since the home has an electric-resistance heating system, the optimal decision variable $h^*(k)$ will be binary (1 or 0) representing an ON or OFF state, respectively. The electricity prices, $c(k)$, are shown in Table II. The off, partial, and peak costs are based on PG&E's weekday summer rates for residential time-of-use schedule E-6 [15]. The morning and evening peak rates were added to make the problem more interesting. For the purposes of simulating the system, we have assumed a heater power, P_H, of 1.5kW and a minimum temperature, T_{min}, of 20 °C.

TABLE II: Residential Time-of-Use Price Schedule

	Time	Cost($/kWh)
Off Peak	12:00AM to 7:00AM & 11:00PM to 12:00AM	0.10376
Morning Peak	7:00AM to 9:00AM	0.25913
Partial Peak	9:00AM to 2:00PM & 9:00PM to 11:00PM	0.18054
Peak	2:00PM to 4:00PM & 6:00PM to 9:00PM	0.29581
Evening Peak	4:00PM to 6:00PM	0.44012

Remark 1 As formulated, it is possible for the MILP to return an infeasible solution error. For example, if $T(0)$ is low and $T(k+1)$ cannot be raised above T_{min} in one time step or if the losses to ambient exceed the energy delivered by the heater for several time steps, then the minimum temperature constraint will be violated and the solver will return no solution. This limitation could be resolved by penalizing the MILP for allowing $T(k+1)$ to fall below T_{min} rather than applying T_{min} as a lower bound constraint. ◁

C. Control Algorithm

The MPC algorithm is executed as follows

1) Set the current temperature measurement as the initial state, $T(0) = T_0$.

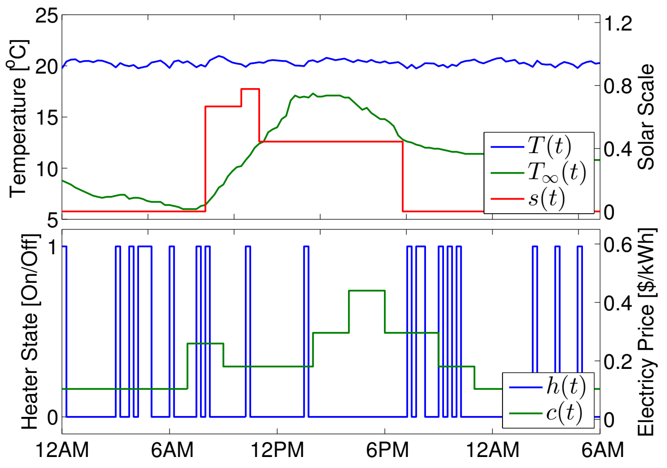

Fig. 7: Deadband Control Simulation

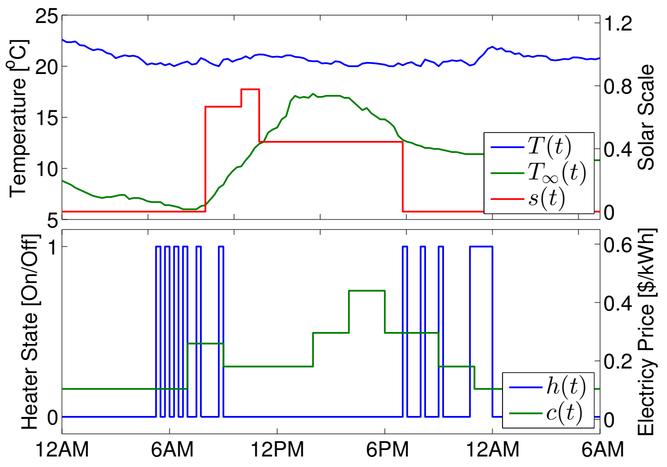

Fig. 8: MPC Simulation

2) Solve the MILP for the optimal open loop input sequence $h^*(0), h^*(1), .., h^*(N-1)$, given forecasts of disturbances $T_\infty(t)$ and $s(t)$.
3) Implement the first input $h^*(0)$ to advance the system one time step.
4) Repeat the algorithm at the next time step.

An advantage of this control algorithm is that a system can be predictively controlled despite inaccuracies in the identified model parameters. In this work, the MILP was solved for a 6 hour time horizon (N=24). It is unlikely that the model accurately predicts the temperature in the apartment 6 hours into the future. Nonetheless, the MILP is capable of estimating the impact of the system's inputs and helping to determine the optimal course of action in the next time step.

D. Simulation Results

A real-time software-in-the-loop simulation was conducted to compare the performance of a traditional deadband controller with our model predictive controller. The identified discrete thermal model with added Gaussian noise was used to represent the evolution of the temperature inside the apartment. The simulation was run for approximately 2

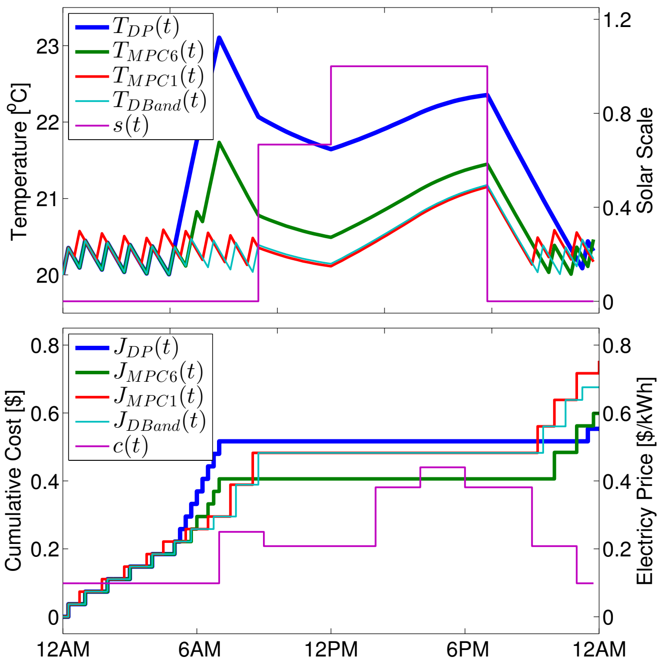

Fig. 9: Simulation Example

weeks using actual weather forecast data, an example of which is shown in Fig. 6. A subset of the results are presented in Fig. 7 and 8.

As shown in the Fig. 7, the deadband controller switches on whenever energy is needed to maintain the desired temperature in the apartment. In contrast, the MPC system avoids peak and partial-peak electricity prices, resulting in the more concentrated periods of heating shown in Fig. 8.

Based on the real time simulation, both the MPC and deadband controllers consume close to the same amount of energy (42.0 kWh and 46.4 kWh, respectively) and maintain comparable average temperatures (21.37 °C and 21.06 °C, respectively) during the 2 weeks studied. However, the MPC system showed a 31% reduction in heating costs compared to the traditional deadband controller. This suggests that the MPC system meets the objective of reducing electricity cost by shifting the time of electricity use rather than decreasing the total energy demand.

While “a 31% reduction in heating costs” is an encouraging finding, it is an insufficient metric for representing the performance of our MPC system. Firstly, it compares MPC with deadband control, which is a trusted but ultimately rudimentary control strategy. By comparing to a strategy that we know to perform poorly, the metric gives an exaggerated impression of success. Secondly, the metric fails to recognize regional or seasonal changes in weather conditions and energy prices. For example, a 5% reduction in cost in a cold climate or during a winter month may be far more significant than a 30% reduction in a temperate climate or during an autumn month. What is needed is a means of defining an optimal control strategy with perfect forecasting. Then, rather than making claims of cost savings, it is possible to express how a system performs relative to the optimal solution.

For the purposes of this work, we have formulated a

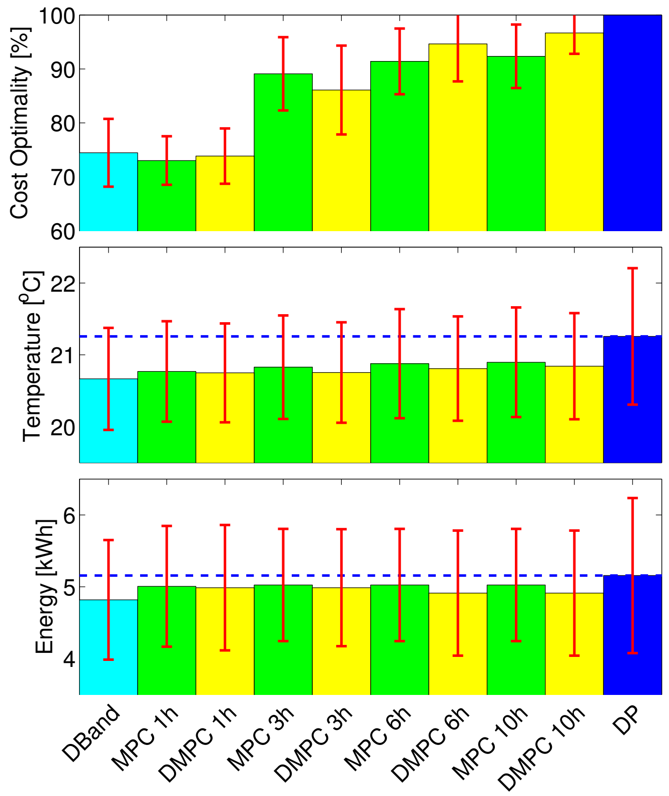

Fig. 10: Mean Cost Optimality, Temperature, and Energy Use per Day over 20 days. Error bars indicate standard deviation over 20 days.

dynamic program (DP) that implements the same objective and system dynamics as the MILP above. Then, we collected weather forecast data for a period of 3 weeks. For each day, we simulated the thermostatic control system with varying degrees of foresight. The deadband controller (DBand) has no forecasting capability and simply turns on the heater for one time step when the lower bound condition is reached. The MPC controller (MPC) was run using the actual (imperfect) weather forecasts at each time step with forecast horizons of 1, 3, 6, and 10 hours. A Deterministic MPC controller (DMPC) was simulated using the recorded conditions as (perfect) weather forecasts with forecast horizons of 1, 3, 6, and 10 hours. Lastly, the DP was simulated to determine the optimal solution to the given objective and compared with the results of the DBand, MPC, and DMPC simulations, as shown in Figures 9 and 10. The cost optimality (CO) for each day simulated is defined as

$$CO(i,j) = \frac{J(DP,j)}{J(i,j)}, \tag{28}$$

where
i = Controller [DBand, MPC, DMPC, DP]
j = Day simulated (out of N=20)
$J(i,j)$ = Cost of heating on day j with controller i

By comparing the results of each controller, we are able to identify some of their behaviours. For example, the MPC and DMPC controllers with 1 hour forecast horizons are not able to see changes in the price of electricity until 1 hour before they take effect. As a result, the controllers are less able to avoid high electricity prices than the DP and, on average, incur higher costs than the non-forecasting DBand controller. For each time horizon, the difference in the cost optimality between the MPC and DMPC simulations indicate the benefit of accurate weather forecasting. By comparing the various forecast horizons of the MPC or DMPC simulations, we can see the diminishing returns of foresight. Additionally, we can see that as the horizon increases, the controller will maintain a slightly higher average temperature so as to minimize heating during on-peak periods.

V. CONCLUSIONS

The results of our work show that the system is capable of a) monitoring the real-time thermal conditions in a space, b) reducing peak loads on the power grid by using electricity price as a proxy for peak demand, c) making use of weather forecast data to predict heating requirements according to the apartment's thermal dynamics, and d) enable smart home technology development with a distributed and cloud-based system architecture. Additionally, this simple MPC control system is capable of reducing the consumer's electricity costs while maintaining a given temperature range.

REFERENCES

[1] U.S. Department of Energy. 2010 Buildings Energy Data Book. Accessed May. 2, 2014. [Online]. Available: http://buildingsdatabook.eren.doe.gov

[2] ——. (2012, June) Electric Resistance Heating. Accessed May. 2, 2014. [Online]. Available: http://energy.gov/energysaver/articles/electric-resistance-heating

[3] J. Wen and T. F. Smith, "Development and validation of online parameter estimation for HVAC systems," *Journal of Solar Energy Engineering*, vol. 125, no. 3, pp. 324–330, 2003.

[4] M. Maasoumy, B. Moridian, M. Razmara, M. Shahbakhti, and A. Sangiovanni-Vincentelli, "Online simultaneous state estimation and parameter adaptation for building predictive control," in *ASME Dynamic Systems and Control Conference*, Palo Alto, California, 2013.

[5] P. Radecki and B. Hencey, "Online building thermal parameter estimation via unscented Kalman filtering," in *American Control Conference*, Montreal, Canada, June 2012.

[6] M. Maasoumy, C. Rosenberg, A. Sangiovanni-Vincentelli, and D. S. Callaway, "Model predictive control approach to online computation of demand-side flexibility of commercial buildings HVAC systems for supply following," in *American Control Conference (ACC)*, Portland, Oregon, 2014, pp. 1082 – 1089.

[7] M. Maasoumy and A. Sangiovanni-Vincentelli, "Total and peak energy consumption minimization of building HVAC systems using model predictive control," *IEEE Design and Test of Computers*, vol. 29, no. 4, pp. 26 – 35, June 2012.

[8] Y. Ma, A. Kelman, A. Daly, and F. Borrelli, "Predictive control for energy efficient buildings with thermal storage: Modeling, simulation, and experiments," *IEEE Control Systems Magazine*, vol. 31, no. 1, pp. 44–64, Feb. 2012.

[9] ZigBee Protocol. [Online]. Available: zigbee.org/Specifications/ZigBee/Overview.aspx

[10] Google App Engine. [Online]. Available: appengine.google.com

[11] Xively Web Service and API. [Online]. Available: xively.com

[12] Heroku App Development Platform. [Online]. Available: heroku.com

[13] Weather Underground Web Service and API. [Online]. Available: wunderground.com/weather/api/

[14] P. Ioannou and J. Sun, *Robust Adaptive Control*. Prentice-Hall, 1996.

[15] Pacific Gas & Electric. Residential Electricity Rates (MAR 1 - APR 30, 2014). Accessed May. 2, 2014. [Online]. Available: http://www.pge.com/tariffs/electric.shtml