2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Reinforcement Learning for Control of Building HVAC Systems

Naren Srivaths Raman, Adithya M. Devraj, Prabir Barooah, and Sean P. Meyn

***Abstract*— We propose a reinforcement learning-based (RL) controller for energy efficient climate control of commercial buildings. Model-based control techniques like model predictive control (MPC) for this problem are challenging to implement as they need simple yet accurate models, which are hard to obtain due to the complexity in hygrothermal dynamics of a building and its HVAC system. RL is an attractive alternative to MPC since once the policy is learned, computing the control in real time involves solving a simple low dimensional optimization problem that does not involve a model of building physics. However, training an RL controller is computationally expensive, and there are many design choices that affect performance.**

We compare in simulations the proposed RL controller, an MPC controller, and a baseline rule-based controller that is widely used in practice. Both the RL and MPC controllers are able to maintain temperature and humidity constraints, and they both reduce energy use significantly compared to the baseline, though the savings by RL is smaller than that by MPC.

I. Introduction

In the U.S., about 19% of the total energy consumption is by commercial buildings [1], of which 44% is used by heating, ventilation, and air conditioning (HVAC) systems [2]. Using advanced climate control algorithms is a cost-effective way to reduce the large energy footprint of HVAC systems. A climate control algorithm must be able to maintain thermal comfort and indoor air quality. It should also use minimal energy. Currently used control algorithms are rule-based and utilize conservatively designed set points, ensuring thermal comfort and indoor air quality, but causing high energy consumption.

In recent years model predictive control (MPC) has emerged as a popular alternative to rule-based control in the academic literature [3]. MPC has the ability to satisfy these competing goals. Despite many simulation—and even some experimental—works showcasing the ability of MPC to significantly reduce energy usage, MPC has not been widely adopted in practice. A challenge with MPC is that simple yet accurate models are needed as it involves real-time optimization. Obtaining such control-oriented models is a challenging task particularly for HVAC systems since the process dynamics of such systems are complex. Another challenge with MPC for buildings is the associated computational complexity. The optimization in MPC is typically non-convex because of the nonlinearities in the hygrothermal dynamics of building and its HVAC system [4–6]. Depending on the planning horizon, the number of decision variables can be large, and solving such a high dimensional non-convex constrained optimization problem in real-time, at every time step, can be challenging.

An alternative approach to MPC is reinforcement learning (RL). RL is a collection of tools used to approximate an optimal policy based on data gathered from a physical system or its simulation. It has two key advantages over MPC:

- It is "model free": no plant model is needed to compute the control decisions in real-time. Although a simulation model is typically needed for off-line learning, unlike MPC, this model can be of arbitrary complexity as it is not used for optimization.
- Despite large computational complexity in the learning phase, the final implementation is a simple state-feedback control at each time step.

As a result of these advantages, along with increase in computing power, application of RL for climate control of buildings is gaining popularity [7–9]. Still, there are several challenges involved in applying RL to commercial buildings. Firstly, the indoor temperature and humidity of the building needs to be maintained within the comfort limits, which is a state constraint, and is hard to impose using model-free techniques such as RL. Secondly, the control inputs such as supply air flow rate, supply air temperature, etc., usually take on continuous values, and have actuator constraints. Therefore, popular RL techniques like Watkins' Q-learning and deep Q-network (DQN) [10], used in prior works [8, 11, 12], cannot be directly applied. Lastly, performance of RL is sensitive to the choice of many design options, such as the cost function and states.

There are only a few prior works on using RL for climate control of commercial buildings; see the review paper [7] and references therein. In [11, 12], the authors use Watkins' Q-learning algorithm which requires discretization of state and action space, and is known to be slow [13]. Some works use the deep reinforcement learning (DRL) technique which can handle large state spaces. DRL is used to control radiant heating system in an office building in [9], while [8] uses DRL for controlling air flow rates. In both works [8, 9] the action space is discretized. For the HVAC system configuration considered in this work, which has four control commands, even a coarse discretization of actions leads to a large number of possible combinations.

Contributions:

(i) We address the challenges of continuous state and action space by applying the Zap Q-learning algorithm [13] with linear function approximation setting. The algorithm is known to have very fast convergence, and moreover, unlike other Q-learning algorithms, it is

This research reported here has been partially supported by the NSF through awards # 1646229 (CPS/ECCS) and # 1934322 (CMMI).

The authors are with the University of Florida, Gainesville, Florida 32611, USA. narensraman@ufl.edu.

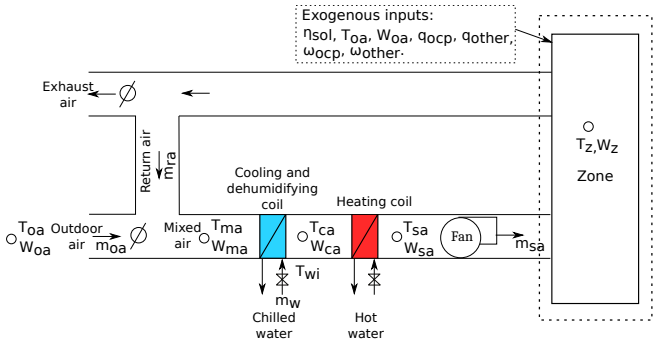


Fig. 1: Schematic of a single-zone commercial variable-air-volume HVAC system.

known to be convergent even in a function approximation setting.

(ii) We provide design guidelines to address some of the challenges mentioned above.

- We design the cost function to minimize the energy, while making sure that the temperature and humidity constraints are not violated during implementation (even with no access to a model).
- We design the state-space so that the actuator constraints are implicitly imposed when implementing a state-feedback controller.

(iii) We compare the performance of the RL controller with an MPC controller proposed in our prior work [6] and a widely used rule-based controller called single maximum [14].

Simulation results show that the proposed RL controller is able to maintain temperature and humidity within the comfort limits, and reduces energy usage when compared to the baseline. However, the energy savings are lower than that provided by the MPC controller.

The rest of this paper is organized as follows. Section II describes the HVAC system considered in this work and the models used in simulating the plant. Section III presents our proposed RL-based controller. Section IV describes the MPC and baseline controllers used for comparison. Simulation results are discussed in Section V. Section VI summarizes the conclusions.

II. System Description and Models

In this paper we consider a single-zone commercial building equipped with a variable-air-volume HVAC system, whose schematic is shown in Figure 1. In such a system, the outdoor air is mixed with part of the air exhausted from the zone. This mixed air is sent through the cooling coil where the air is cooled and dehumidified to conditioned air temperature (T_{ca}) and humidity ratio (W_{ca}). Then the conditioned air is reheated as needed and finally supplied to the zone. Note that there is only change in temperature across the reheat coil, the humidity remains constant, i.e., $W_{sa} = W_{ca}$, where W_{sa} is the humidity ratio of the supply air.

There are 4 control commands a climate control system needs to decide. They are: (i) supply air flow rate (m_{sa}), (ii) outdoor air ratio ($r_{oa} := \frac{m_{oa}}{m_{sa}}$, where m_{oa} is the outdoor air flow rate), (iii) conditioned air temperature (T_{ca}), and (iv) supply air temperature (T_{sa}). So the control command/input vector is, $u(t) := [m_{sa}(t), r_{oa}(t), T_{ca}(t), T_{sa}(t)]^T \in \Re^4$. These inputs need to be varied in such a way that the thermal comfort and indoor air quality are maintained in the zone, while using minimal energy.

In order to compare the performance of the control algorithms presented and to train the proposed RL controller, we need simulation models of the zone's temperature and humidity dynamics, certain HVAC system components, and power consumed by the HVAC system. These models are presented in detail in our prior work [6]. We present only the relevant salient features below.

A. Humidity and Temperature Dynamic Model

For the thermal dynamics of the zone, we use a resistor-capacitor (R-C) model, specifically a 2R-2C model with the two states being the zone temperature and wall temperature. The humidity dynamic is modeled based on mass balance. The overall model is of the form:

$$\dot{x}(t) = f\big(x(t), u(t), v(t), w(t)\big), \tag{1}$$

where the state vector $x(t)$ consists of zone temperature (T_z), wall temperature (T_w), and humidity ratio of the zone (W_z), i.e., $x(t) := [T_z(t), T_w(t), W_z(t)]^T \in \Re^3$. The input vector $u(t)$ is defined above. The internal variable $v(t)$ consists of conditioned air humidity ratio, $v(t) := W_{ca}(t) \in \Re^1$. The exogenous input vector $w(t)$ consists of outdoor air temperature (T_{oa}), outdoor air humidity ratio (W_{oa}), solar irradiance (η_{sol}), internal heat load (q_{ocp}) and rate of internal water vapor generation (ω_{ocp}) due to people, and internal heat load (q_{other}) and rate of internal water vapor generation (ω_{other}) due to other sources like equipment, i.e., $w(t) := [T_{oa}(t), W_{oa}(t), \eta_{sol}(t), q_{ocp}(t), \omega_{ocp}(t), q_{other}(t), \omega_{other}(t)]^T \in \Re^7$.

B. Cooling and Dehumidifying Coil Model

The cooling coil model consists of five inputs and two outputs. The inputs are supply air flow rate (m_{sa}), chilled water flow rate (m_w), chilled water temperature (T_{wi}), mixed air temperature (T_{ma}) and humidity ratio (W_{ma}). The outputs are conditioned air temperature (T_{ca}) and humidity ratio (W_{ca}). We use a gray box data-driven model which was developed in [6]. The parameters of this model were fit based on data collected from EnergyPlus [15].

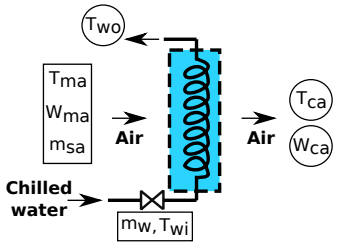


Fig. 2: Cooling and dehumidifying coil, and relevant variables (model inputs in rectangles, outputs in circles).

C. Power Consumption Models

There are three major components which consume power in the HVAC system considered in this work: fan, cooling coil, and reheat coil. We assume that the power consumed by other components such as damper motors is negligible.

The fan power consumption P_{fan} is modeled as being proportional to a quadratic in the supply air flow rate. The cooling coil power consumption is modeled as being proportional to the heat extracted from the mixed air stream and is of the form: $P_{cc}(t) = g_{cc}\big(m_{sa}(t), h_{ma}(t), h_{ca}(t)\big)$, where h_{ma} is the enthalpy of the mixed air and h_{ca} is the enthalpy of the conditioned air.

The reheat coil power consumption is modeled as being proportional to the heat added to the conditioned air stream and is of the form: $P_{reheat}(t) = g_{reheat}\big(m_{sa}(t), T_{sa}(t), T_{ca}(t)\big)$.

III. Reinforcement Learning-Based Controller

A. Markov Decision Process (MDP) Formulation

Reinforcement learning (RL) is a collection of tools used to approximate an optimal policy based on data gathered from a physical system or its simulation. In our application, RL is used to learn a control policy that minimizes energy consumption of a commercial building HVAC system, while ensuring that the zone temperature and humidity are within the desired comfort limits.

To this end, we model the problem as a discrete-time Markov decision process comprised of the tuple $(\mathcal{X}, \mathcal{U}, \mathcal{P}, c, \beta)$, where $\mathcal{X}$ denotes the state-space, $\mathcal{U}$ denotes the action-space (a set of all *feasible* actions), $\mathcal{P}$ denotes the transition kernel, $c : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \to \Re$ denotes the cost function, and β denotes a discount factor.

Since the underlying system described in Section II is evolving in continuous time, we discretize the time-steps for our MDP formulation, with a sampling interval $\Delta t > 0$. We denote by $X_k \in \mathcal{X}$ the state, and $U_k \in \mathcal{U}$ the control input at time-instant $k\Delta t$ (or simply, time-step k).

The goal is to obtain a state-feedback policy $\phi^* : \mathcal{X} \to \mathcal{U}$ that minimizes the sum of expected discounted cost:

$$\phi^* := \underset{\phi:\mathcal{X}\to\mathcal{U}}{\operatorname{argmin}} \left\{ \sum_{k=0}^{\infty} \beta^k \mathsf{E}\big[c(X_k, U_k, X_{k+1})\big] \right\} \tag{2}$$

with $U_k = \phi(X_k)$ for $k \geq 0$.

There are several design choices that need to be made when formulating the underlying problem as an MDP. Here, we discuss some of the choices we made in our formulation, and the intuition behind them.

Choosing the state-action space:

Based on the system model defined in Section II, we define:

$$U_k := [m_{sa}(k), r_{oa}(k), T_{ca}(k), T_{sa}(k)]^T. \tag{3}$$

The choice of X_k is more subtle. Based on the discussion in Section II-A, it is natural to include the zone temperature $T_z(k)$, wall temperature $T_w(k)$, and zone humidity $W_z(k)$. However, since the wall temperature is only an auxiliary state that is not directly measurable, we do not include it as a part of X_k.

In addition to the observed states T_z and W_z, we also make certain exogenous inputs part of X_k, since they can provide valuable information. Among the various exogenous inputs defined in Section II-A, only outdoor air temperature T_{oa}, outdoor air humidity W_{oa}, and solar irradiance η_{sol} are measured. Of these, we choose to include T_{oa} and W_{oa} in the state for the following reasons:

- Since solar irradiance is highly correlated with the outdoor air temperature, making it a part of the system state does not provide much additional useful information.
- In many cases, buildings are not equipped with a solar irradiance sensor.

Lastly, we include U_{k-1}, the control inputs in the previous time-step to be part of X_k. This is required to impose rate constraints in a state-feedback policy. We therefore have:

$$\begin{aligned} X_k := [&T_z(k), W_z(k), T_{oa}(k), W_{oa}(k), \\ &m_{sa}(k-1), r_{oa}(k-1), T_{ca}(k-1), T_{sa}(k-1)]^T \end{aligned} \tag{4}$$

Feasible actions/control inputs:

At each time-step k, the input U_k needs to satisfy the following constraints that are imposed due to actuator limits, ventilation requirements, etc.

$$\begin{aligned} \max\big(m_{sa}(k-1) - m_{sa}^{rate}\Delta t, m_{sa}^{low}\big) &\leq m_{sa}(k) \\ &\leq \min\big(m_{sa}(k-1) + m_{sa}^{rate}\Delta t, m_{sa}^{high}\big) \end{aligned} \tag{5a}$$

$$\begin{aligned} \max\big(r_{oa}(k-1) - r_{oa}^{rate}\Delta t, r_{oa}^{low}\big) &\leq r_{oa}(k) \\ &\leq \min\big(r_{oa}(k-1) + r_{oa}^{rate}\Delta t, r_{oa}^{high}\big) \end{aligned} \tag{5b}$$

$$\begin{aligned} \max\big(T_{ca}(k-1) - T_{ca}^{rate}\Delta t, T_{ca}^{low}\big) &\leq T_{ca}(k) \\ &\leq \min\big(T_{ca}(k-1) + T_{ca}^{rate}\Delta t, T_{ma}(k)\big) \end{aligned} \tag{5c}$$

$$\begin{aligned} \max\big(T_{sa}(k-1) - T_{sa}^{rate}\Delta t, T_{ca}(k)\big) &\leq T_{sa}(k) \\ &\leq \min\big(T_{sa}(k-1) + T_{sa}^{rate}\Delta t, T_{sa}^{high}\big) \end{aligned} \tag{5d}$$

If U_k satisfies the above constraints, we say $U_k \in \mathcal{U}(X_k)$.

Constraint (5a) is imposed to take into account the capabilities of the fan. The minimum allowed value for the supply air flow rate is computed based on the ventilation requirements specified in ASHRAE 62.1 [16], as well as to maintain positive building pressurization. This is computed as follows: $m_{sa}^{low} = \max\big\{(m_{oa}^p n_p + m_{oa}^A A)/r_{oa}, m_{oa}^{bp}/r_{oa}\big\}$, where m_{oa}^p is the outdoor air flow rate required per person, n_p is the number of people, m_{oa}^A is the outdoor air required per zone area, A is the zone area, m_{oa}^{bp} is the outdoor air rate required to maintain positive building pressurization, and r_{oa} is the outdoor air ratio.

Constraints (5b)-(5d) take into account the capabilities of the damper actuators, cooling and reheat coils. In constraint (5c), T_{ma} is the mixed air temperature computed as: $T_{ma}(k) = r_{oa}(k)T_{oa}(k) + (1 - r_{oa}(k))T_z(k)$. The inequality $T_{ca}(k) \leq T_{ma}(k)$ ensures that the cooling coil can only cool the mixed air stream. Similarly, the inequality $T_{sa}(k) \geq T_{ca}(k)$ ensures that the reheat coil can only add heat.

Cost function design:

A climate control system has to keep overall energy consumption low while maintaining temperature and humidity of the zone within a desired range. The cost function is designed to capture both these features.

Denote by $\mathcal{X}_{\text{de}}$ the set of all desirable states: at time step $k \geq 0$, we say $X_k \in \mathcal{X}_{\text{de}}$, if the zone temperature and humidity are within desirable limits: $T_z^{low} \leq T_z(k) \leq T_z^{high}$ and $W_z^{low} \leq W_z(k) \leq W_z^{high}$, for given values of T_z^{low} T_z^{high}, W_z^{low}, and W_z^{high}. The cost function is then defined as,

$$c(X_k, U_k, X_{k+1}) = \begin{cases} E_k, & \text{if } X_{k+1} \in \mathcal{X}_{\text{de}} \\ p_{low}, & \text{if } X_k \in \mathcal{X}_{\text{de}} \ \& \ X_{k+1} \notin \mathcal{X}_{\text{de}} \\ p_{high}, & \text{if } X_k \notin \mathcal{X}_{\text{de}} \ \& \ X_{k+1} \notin \mathcal{X}_{\text{de}} \end{cases}$$

where E_k denotes the energy consumed:

$$E_k := E_{fan}(k) + E_{cc}(k) + E_{reheat}(k), \tag{6}$$

where $E_{fan}(k)$, $E_{cc}(k)$, $E_{reheat}(k)$ are the energy consumed by the fan, cooling coil, and reheat coil, respectively, and $p_{high} > p_{low}$ are the penalties imposed for violating the comfort limits.

An occasional violation of limits on T_z or W_z may not be noticed by occupants; a continuous violation can cause discomfort and even lead to serious adverse effects such as mold. To discourage this behavior, we impose the higher penalty p_{high} ($> p_{low}$), when the T_z or W_z constraints are violated over two consecutive time steps.

B. Value Function Approximation and Zap Q-Learning

Under the assumption that the underlying problem is an MDP, it is known that the optimal policy in (2) satisfies:

$$\phi^*(x) = \min_{u \in \mathcal{U}(x)} Q^*(x,u)\,, \qquad x \in \mathcal{X} \tag{7}$$

where $Q^* : \mathcal{X} \times \mathcal{U} \to \Re$ denotes the associated optimal Q-value function:

$$Q^*(x,u) := \min_{\{U_k\}} \sum_{k=0}^{\infty} \beta^k \mathsf{E}\big[c(X_k, U_k, X_{k+1}) | X_0 = x, U_0 = u\big]$$

where the minimization is over all feasible inputs. It is known that Q^* solves the Bellman equation:

$$\begin{aligned} Q^*(x,u) := E\Big[c(X_k, U_k, X_{k+1}) \mid X_k = x, U_k = u\Big] \\ + \beta E\Big[\underline{Q}^*(X_{k+1}) \mid X_k = x, U_k = u\Big] \end{aligned} \tag{8}$$

where, for any $Q : \mathcal{X} \times \mathcal{U} \to \Re$, we use the notation:

$$\underline{Q}(x) := \min_{u \in \mathcal{U}(x)} Q(x,u).$$

Reinforcement learning algorithms such as Q-learning can be used to estimate an approximation for the Q-function, but the theory is weak for the general function approximation setting. Moreover, much of the literature considers a setting wherein the state and action spaces are discretized [11, 12], due to the lack of existing RL algorithms that are capable of dealing with the complex setting.

In this work, we apply the Zap Q-learning of [13] to approximate Q^* using a parameterized family of functions $\{Q^\theta : \theta \in \Re^d\}$. Specifically, we consider linear parameterization, so that for each $x \in \mathcal{X}$ and $u \in \mathcal{U}$,

$$Q^\theta(x,u) = \theta^T \psi(x,u), \tag{9}$$

where $\psi : \mathcal{X} \times \mathcal{U} \to \Re^d$ denotes the "basis functions". We choose the parameterized family to be a set of all quadratic functions in (x,u), leading to basis functions $\{\psi_i\}$ defined in Table I. Once the basis functions are fixed, the Zap Q-learning algorithm defined below can be used to estimate Q^* using the approximation Q^{θ^*}.

Algorithm 1 Zap Q-learning with ε-greedy exploration

Input: $\theta_0 \in \Re^d$, $\hat{A}_0 \in \Re^{d \times d}$, $X_0 \in \mathcal{X}$, $k = 0$, $T \in \mathcal{Z}^+$, $\rho \in (0.5, 1)$, $\varepsilon \in (0,1)$

1: **repeat**

2: With prob. ε, choose $U_k \in \mathcal{U}(X_k)$ uniformly at random, and with prob. $1-\varepsilon$, choose: $U_k = \underset{u \in \mathcal{U}(X_k)}{\text{argmin}}\ Q^{\theta_k}(X_k, u)$

3: Sample next state X_{k+1} with current state X_k and input U_k

4: $\phi_k(X_{k+1}) := \underset{u \in \mathcal{U}(X_{k+1})}{\text{argmin}}\ Q^{\theta_k}(X_{k+1}, u)$;

5: $d_{k+1} := c(X_k, U_k, X_{k+1}) + \beta Q^{\theta_k}(X_{k+1}, \phi_k(X_{k+1})) - Q^{\theta_k}(X_k, U_k)$; ▷ Temporal difference

6: $A_{k+1} := \psi(X_k, U_k)\big[\beta\psi(X_{k+1}, \phi_k(X_{k+1})) - \psi(X_k, U_k)\big]^T$;

7: $\hat{A}_{k+1} = \hat{A}_k + \gamma_k\big[A_{k+1} - \hat{A}_k\big]$; ▷ Matrix gain update

8: $\theta_{k+1} = \theta_k - \alpha_k \hat{A}_{k+1}^{-1} \psi(X_k, U_k) d_{k+1}$; ▷ Zap-Q update

9: $k = k + 1$

10: **until** $k \geq T$

In the above algorithm, the step-size sequences $\{\alpha_k\}$ and $\{\gamma_k\}$ are chosen to be:

$$\alpha_k = (k+1)^{-1} \quad \text{and} \quad \gamma_k = \alpha_k^\rho\,, \qquad k \geq 0$$

where $\rho \in (0.5, 1)$ is a hyper-parameter. The input sequence defined in line 2 is known as "ϵ-greedy" exploration.

The use of Zap-Q with linear function approximation (specifically, the quadratic basis) has several advantages over existing techniques:

(i) Contrary to existing Q-learning algorithms, it is argued in [13] that Zap-Q can converge even in a function approximation setting; this is due to a novel *matrix gain* technique that is capable of stabilizing a potentially unstable recursion.

(ii) Algorithms such as DQN [10], though they use neural-networks to approximate the Q-function, can only be applied to a finite action space setting; this is due to the fact that minimization of a continuous function of the actions (a necessary step in the Q-learning algorithm), when the function is the output of a neural network, can be computationally very expensive [8, 9]. Our formulation on the other hand does not have these issues.

TABLE I: Basis functions $\psi_i^k := \psi_i(X_k, U_k)$ at iteration k.

$\psi_1^k = 1$	$\psi_2^k = T_z(k)$	$\psi_3^k = W_z(k)$
$\psi_4^k = T_{oa}(k)$	$\psi_5^k = W_{oa}(k)$	$\psi_6^k = m_{sa}(k)$
$\psi_7^k = r_{oa}(k)$	$\psi_8^k = T_{ca}(k)$	$\psi_9^k = T_{sa}(k)$
$\psi_{10}^k = (\psi_2^k)^2$	$\psi_{11}^k = (\psi_3^k)^2$	$\psi_{12}^k = (\psi_4^k)^2$
$\psi_{13}^k = (\psi_5^k)^2$	$\psi_{14}^k = (\psi_6^k)^2$	$\psi_{15}^k = (\psi_7^k)^2$
$\psi_{16}^k = (\psi_8^k)^2$	$\psi_{17}^k = (\psi_9^k)^2$	$\psi_{18}^k = \psi_2^k\psi_3^k$
$\psi_{19}^k = \psi_2^k\psi_4^k$	$\psi_{20}^k = \psi_2^k\psi_5^k$	$\psi_{21}^k = \psi_2^k\psi_6^k$
$\psi_{22}^k = \psi_2^k\psi_7^k$	$\psi_{23}^k = \psi_2^k\psi_8^k$	$\psi_{24}^k = \psi_2^k\psi_9^k$
$\psi_{25}^k = \psi_3^k\psi_4^k$	$\psi_{26}^k = \psi_3^k\psi_5^k$	$\psi_{27}^k = \psi_3^k\psi_6^k$
$\psi_{28}^k = \psi_3^k\psi_7^k$	$\psi_{29}^k = \psi_3^k\psi_8^k$	$\psi_{30}^k = \psi_3^k\psi_9^k$
$\psi_{31}^k = \psi_4^k\psi_5^k$	$\psi_{32}^k = \psi_4^k\psi_6^k$	$\psi_{33}^k = \psi_4^k\psi_7^k$
$\psi_{34}^k = \psi_4^k\psi_8^k$	$\psi_{35}^k = \psi_4^k\psi_9^k$	$\psi_{36}^k = \psi_5^k\psi_6^k$
$\psi_{37}^k = \psi_5^k\psi_7^k$	$\psi_{38}^k = \psi_5^k\psi_8^k$	$\psi_{39}^k = \psi_5^k\psi_9^k$
$\psi_{40}^k = \psi_6^k\psi_7^k$	$\psi_{41}^k = \psi_6^k\psi_8^k$	$\psi_{42}^k = \psi_6^k\psi_9^k$
$\psi_{43}^k = \psi_7^k\psi_8^k$	$\psi_{44}^k = \psi_7^k\psi_9^k$	$\psi_{45}^k = \psi_8^k\psi_9^k$

(iii) *Assuming* that competing algorithms are convergent, the Zap-Q algorithm is known to be orders of magnitude faster than existing techniques in terms of sample efficiency [13].

C. Off-Line Learning and Real-Time Control

Algorithm 1 is implemented on a simulation model of the building. Once we approximate $Q^*(x,u)$ using $Q^{\theta_T}(x,u)$, with θ_T denoting the parameter vector estimate obtained from the algorithm after T iterations, the online state-feedback control can be obtained using (7), with Q^* replaced by Q^{θ_T}. Notice that the optimization problem in (7) is straightforward to solve; from (9) we can see that the cost function is quadratic in u, with box constraints (5a)-(5d), and just 4 decision variables.

IV. Control Algorithms Used for Comparison

We compare the performance of the RL controller with two others, an MPC controller and a rule-based controller. These are briefly described below. The interested reader is referred to [6] for a detailed description of the MPC controller.

A. Model Predictive Controller

This controller was proposed in our prior work [6]. The goal of this MPC controller is the same as the proposed RL controller: minimize energy use while maintaining temperature and humidity constraints. The control inputs are computed in discrete time steps Δt, for a finite planning horizon N, by solving a constrained optimization problem. There are three pieces of information needed for solving this problem. They are: (i) the current value of the states, (ii) prediction of the exogenous inputs mentioned in Section II-A over the planning horizon, and (iii) model of the building and HVAC system. The control inputs for the first time step, obtained from solving this problem, are applied to the plant. At the next time step this process is repeated.

The states for this MPC controller are the zone temperature and humidity, $x(k) = [T_z(k), W_z(k)]^T$. The control inputs are the same as those mentioned in Section II. The optimization problem solved is to minimize the total energy consumption—fan, cooling coil, and reheat coil—over a planning horizon N, subject to: (i) the various control input constraints—(5a)-(5d)—mentioned in Section III, (ii) equality constraints due to model of the temperature and humidity dynamics of the zone, (iii) box constraints to maintain temperature and humidity of the zone within the allowed comfort limits, (iv) equality constraints due to model of the cooling and dehumidifying coil. *Unlike RL, state constraints are explicitly imposed, as MPC has models to compute the state evolution/trajectory.*

B. Baseline Controller

For the baseline, we consider the widely used rule-based controller called single maximum [14]. This controller operates in three modes based on the zone temperature: cooling, deadband, and reheating. The supply air flow rate (m_{sa}) and supply air temperature (T_{sa}) are varied based on the mode the controller is in.

The conditioned air temperature (T_{ca}) is typically kept constant at a low value ($55°F$), which ensures that the air supplied to the zone is dry enough at all times. The outdoor air ratio (r_{oa}) is varied to maintain the ventilation requirements dictated by ASHRAE 62.1 [16] and the positive building pressurization requirements. The zone temperature set points vary based on occupied/unoccupied hours.

V. Simulation Results and Discussions

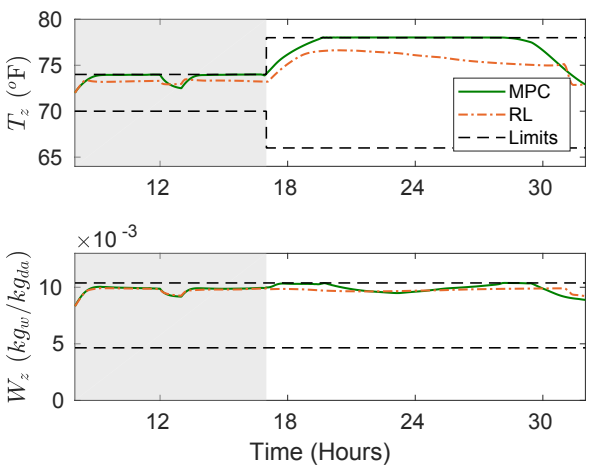
A. Simulation Parameters

The parameters for the building and HVAC system models, presented in Section II, are chosen to mimic an auditorium in Pugh hall (5000 sq.ft.), located in the University of Florida campus. Due to lack of space we present only the relevant details here, the interested readers are referred to [6].

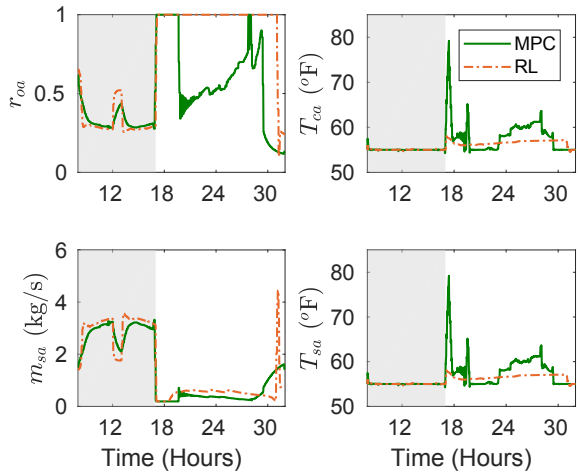
The scheduled hours of occupancy are 8:00 AM to 5:00 PM, during which the following temperature and humidity constraints are used: $T_z^{low,occ} = 294.3\ K\ (70\ °F)$, $T_z^{high,occ} = 296.5\ K\ (74\ °F)$, $W_z^{low,occ} = 0.0046\ kg_w/kg_{da}$, and $W_z^{high,occ} = 0.0104\ kg_w/kg_{da}$. The unoccupied hours are between 5:00 PM to 8:00 AM, during which the constraints are relaxed to: $T_z^{low,unocc} = 292\ K\ (66\ °F)$, $T_z^{high,unocc} = 298.7\ K\ (78\ °F)$, $W_z^{low,unocc} = 0.0046\ kg_w/kg_{da}$, and $W_z^{high,unocc} = 0.0104\ kg_w/kg_{da}$.

RL parameters: The various parameters used in the RL controller are listed in Table II.

The constraints on zone temperature are more relaxed during the unoccupied mode when compared to the occupied mode, i.e., $[T_z^{low,occ}, T_z^{high,occ}] \subseteq [T_z^{low,unocc}, T_z^{high,unocc}]$. Let's say T_z is between $T_z^{high,occ}$ and $T_z^{high,unocc}$ at a given time step. Depending on the mode, the zone is warmer than the allowed limit or not. Since distinct control decisions are expected in situations like this, we use different parameters in the Q function for the occupied and unoccupied modes. So the number of parameters (θ) in the Q function are: $2d = 2 \times 45 = 90$.



(a) Zone conditions with the black dashed lines showing the upper and lower comfort limits (zone air temperature and zone air humidity ratio).



(b) Control commands/inputs (outdoor air ratio, conditioned air temperature, supply air flow rate, and supply air temperature).

Fig. 3: Comparison of RL and MPC controllers for a hot-humid day (September/07/2016) and 100% occupancy level. The scheduled hours of occupancy are shown as the gray shaded area. Outdoor weather data obtained from nsrdb.nrel.gov for Gainesville, FL.

TABLE II: RL parameters.

β	ρ	p_{low}	p_{high}
0.95	0.8	15	20
m_{sa}^{rate}	T_{ca}^{rate}	T_{sa}^{rate}	r_{oa}^{rate}
$(kg/s/min)$	(K/min)	(K/min)	$(\%/min)$
0.37	0.56	0.56	6
m_{sa}^{high}	T_{ca}^{low}	T_{sa}^{high}	r_{oa}^{low}
(kg/s)	(K)	(K)	$(\%)$
4.6	285.9	303.2	0
m_{oa}^{p}	m_{oa}^{A}	m_{oa}^{bp}	r_{oa}^{high}
$(kg/s/person)$	$(kg/s/m^2)$	(kg/s)	$(\%)$
0.0043	3.67×10^{-4}	0.1894	100

The simulations to learn the parameters of the Q function are performed using MATLAB. Three months of summer weather data, between June/06 to September/05 of 2016, for Gainesville, Florida, is obtained from the National Solar Radiation Database (nsrdb.nrel.gov). The parameters of the Q function for the occupied and unoccupied modes are obtained through separate simulations. We use 100% design occupancy of 175 people (n_p^{design}) for the implementation of Algorithm 1 in the occupied mode. For the unoccupied mode, zone has no occupants in our application of Algorithm 1.

We warm start the training simulations by using the *Kalman filter* matrix gain of [17] for the first 2×10^5 iterations after which the ZAP gain ($\hat{A}$) is used.

MPC parameters: We used a planning horizon of 24 hours, with sampling period of 5 minutes, resulting in $N = 288$. The underlying optimization problem has 2304 decision variables. The rate constraint values are the same as those presented for the RL controller.

Baseline controller parameters: The cooling and reheating set points are chosen to be 296.5 K (74 $^\circ F$) and 294.3 K (70 $^\circ F$) respectively, during the occupied mode. For the unoccupied mode, the cooling and reheating set points are 298.7 K (78 $^\circ F$) and 292 K (66 $^\circ F$) respectively. The conditioned air temperature is kept at 285.9 K (55 $^\circ F$).

Computational complexity: The optimization problem in RL and MPC is solved using CasADi [18] and IPOPT [19], a nonlinear programming (NLP) solver, on a Desktop computer running Linux with 16 GB RAM and a 3.60GHz $\times$ 8 CPU. Note that the number of decision variables for RL is only 4 while for MPC it is 2304. On an average it takes 0.01 seconds to solve the optimization problem for RL and 2 seconds for MPC. The parameters of the Q function are seen to settle after 2.5×10^6 iterations and takes about 42 hours to train.

B. Results and Discussions

We now compare the performance of the RL controller, the MPC controller, and the baseline controller through simulations. Typically, in auditoriums/lecture classrooms, the actual occupancy level is lower than the design value. We test the performance of the controllers under such conditions. This is to examine if the controllers are robust to mismatch between the design conditions and reality. Four levels of occupancy are tested: 25%, 50%, 75%, and 100%. The simulations are done for 24 hours starting from 8:00 AM. The occupancy pattern considered in the plant is: the zone is occupied between 8:00 AM to 12:00 PM and 1:00 PM to 5:00 PM.

For each of the three controllers we assume the following: (i) the number of occupants is not measured, (ii) the zone is occupied throughout the scheduled hours of occupancy, between 8:00 AM to 5:00 PM, shown as the gray shaded area in Figure 3. So the controllers need to ensure that the outdoor air needed to satisfy the ventilation requirements corresponding to the designed number of occupants (175), is provided during these scheduled hours of occupancy, according to ASHRAE 62.1.

The MPC controller requires prediction of the exogenous inputs for the planning horizon N. We compute the occupant induced heat load (q_{ocp}) and rate of internal water vapor generation due to people (ω_{ocp}), based on the above mentioned occupancy pattern and designed number of occupants (175). The remaining exogenous inputs are assumed fully known.

For all the four levels of occupancy, simulation results show that compared to the baseline, the proposed RL controller reduces energy use by $\sim 20\%$ while the MPC controller reduces by $\sim 30\%$. The high energy use of the baseline can be attributed to two main factors. First, the minimum allowed value of the supply air flow rate needs to be high enough so that it is able to maintain the design heating load with a low enough supply air temperature to prevent stratification. But this leads to high energy use [14]. Second, the conditioned air temperature is kept at a low value in the interest of humidity, but such a low value is not always necessary.

Figure 3 shows the simulation results for a hot and humid day, Sep/07/2016, with 100% occupancy level. For the sake of clarity we do not present the baseline in the figure. It can be seen from Figure 3(a) that both the RL and MPC controllers are able to maintain zone temperature and humidity within the comfort limits (shown as the dashed black lines). Note that for the MPC controller, the temperature and humidity constraints are active most of the time. This behavior can be interpreted as the MPC controller trying to keep the zone as warm and humid as is allowed to save energy, especially considering that it is a hot and humid day. Such behavior is not observed in the RL controller. Since the RL controller is model-free, the state constraints cannot be explicitly imposed. Rather it is learnt from the penalties imposed due to state constraint violation during the learning process. This could be one of the reasons why the RL controller does not save as much energy as MPC.

The RL controller is found to be robust to lack of occupancy information (results not shown due to lack of space). It is able to maintain the temperature and humidity at all four levels of occupancy even though it is trained assuming 100% occupancy.

VI. Conclusion

An RL-based controller is presented for commercial HVAC systems. Unlike prior work in this area, the proposed controller does not need discretization of action space; instead it works with continuous state and action spaces. The controller uses information that is readily available in most modern commercial buildings. Simulation results show that the proposed RL controller is able to maintain temperature and humidity within the comfort limits, while reducing energy use compared to a baseline controller. MPC performs slightly better than RL in terms of energy use. We believe that combining RL with MPC could lead to better performance. We plan to explore this in the future.

References

[1] U.S. DoE, "Buildings energy data book," *Energy Efficiency & Renewable Energy Department*, 2011.

[2] "Commercial buildings energy consumption survey (CBECS): Overview of commercial buildings, 2012," Energy information administration, Department of Energy, U.S. Govt., Tech. Rep., December 2012. [Online]. Available: https://www.eia.gov/consumption/commercial/reports/2012/energyusage/

[3] G. Serale, M. Fiorentini, A. Capozzoli, D. Bernardini, and A. Bemporad, "Model predictive control (MPC) for enhancing building and HVAC system energy efficiency: Problem formulation, applications and opportunities," *Energies*, vol. 11, no. 3, p. 631, 2018.

[4] E. Atam and L. Helsen, "A convex approach to a class of non-convex building HVAC control problems: Illustration by two case studies," *Energy and Buildings*, vol. 93, pp. 269 – 281, 2015.

[5] T. Zeng and P. Barooah, "An autonomous mpc scheme for energy-optimal control of building HVAC systems," in *American Control Conference*, July 2020, accepted.

[6] N. S. Raman, K. Devaprasad, and P. Barooah, "MPC-based building climate controller incorporating humidity," in *American Control Conference*, July 2019, pp. 253–260.

[7] K. Mason and S. Grijalva, "A review of reinforcement learning for autonomous building energy management," *arXiv.org*, 2019, arXiv:1903.05196.

[8] T. Wei, Y. Wang, and Q. Zhu, "Deep reinforcement learning for building HVAC control," in *Proceedings of the 54th Annual Design Automation Conference 2017*, ser. DAC '17. New York, NY, USA: ACM, 2017, pp. 22:1–22:6.

[9] Z. Zhang and A. Chong, "A deep reinforcement learning approach to using whole building energy model for HVAC optimal control," in *Building Performance Modeling Conference and SimBuild*, 2018.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[11] S. Liu and G. P. Henze, "Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory: Part 1. Theoretical foundation," *Energy and Buildings*, vol. 38, no. 2, pp. 142 – 147, 2006.

[12] B. Li and L. Xia, "A multi-grid reinforcement learning method for energy conservation and comfort of HVAC in buildings," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2015, pp. 444–449.

[13] A. M. Devraj and S. Meyn, "Zap Q-learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 2235–2244.

[14] ASHRAE, "The ASHRAE handbook : Applications (SI Edition)," 2011.

[15] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte, and J. Glazer, "EnergyPlus: creating a new-generation building energy simulation program," *Energy and buildings*, vol. 33, no. 4, pp. 319–331, 2001.

[16] ASHRAE, "ANSI/ASHRAE standard 62.1-2016, ventilation for acceptable air quality," 2016.

[17] D. Choi and B. Van Roy, "A generalized kalman filter for fixed point approximation and efficient temporal-difference learning," *Discrete Event Dynamic Systems*, vol. 16, no. 2, pp. 207–239, 2006.

[18] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, Jul 2018, , first available online.

[19] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar 2006.