REINFORCEMENT LEARNING AND REAL-TIME BUILDING THERMAL PERFORMANCE DATA VISUALIZATION

RAVI S. SRINIVASAN, ALI M. MALKAWI
Department of Architecture, School of Design
University of Pennsylvania, Philadelphia, PA, USA.
{sravi, malkawi}@design.upenn.edu

Abstract. Computational Fluid Dynamics (CFD) simulations are used to predict the fluid behaviour and particle systems-in-action in three-dimensional space, allowing experts to evaluate a series of environmental decisions in designing buildings. Although computing power has increased in the past decade, detailed CFD simulations introduce time-delay that defeats the notion of real-time data visualization. A method that can bypass the time-consuming simulations and generate results comparable to detailed CFD simulations will allow such visualizations to be constructed. This paper discusses a pilot project that utilizes a Reinforcement Learning (RL) algorithm coupled with a simplified fluid dynamics equation to generate thermal performance data for real-time visualization.

1. Introduction

Computational Fluid Dynamics applies numerical techniques to solve the conservation equations for mass, momentum, and thermal energy for fluid fields. These simulations provide accurate data on flow movements and behaviours of complex systems, and have been widely used in many applications in relation to buildings, including: natural ventilation design (Carrilho-da-Graca et al., 2002), prediction of smoke and fire (Yeoh et al., 2003), building material emissions for indoor air quality assessment (Huang and Haghighat, 2002), and indoor environmental analysis (Eckhardt and Zori, 2002).

Visualization of these simulation datasets has evolved from static two-dimensional to dynamic three-dimensional analyses, such as in virtual environments – both Virtual Reality (VR) and Augmented Reality (Wasfy and Noor, 2002; Malkawi and Srinivasan, 2005). In spite of significant advances in CFD visualization techniques and computer hardware, real-time data visualization remains a challenge. This is primarily due to the time delay associated with solving complex heat-mass transfer equations for fluid fields. Several research projects were conducted to enable rapid explorations of post-processed data, such as parallelized computing (Lane, 1995), out-of-core particle tracing (Ueng et al., 1997), load balancing (Bajaj et al., 1999), and immersive data visualization (Malkawi and Srinivasan, 2005). Despite

the efforts of these researchers, these techniques cannot be employed for real-time building thermal performance data visualization. One approach to solve this problem is employing machine-learning algorithms coupled with approximate fluid field study techniques to bypass time-consuming detailed CFD simulations, as depicted in Figure 1. Such an approach will allow the generation of performance datasets for real-time visualization applications.

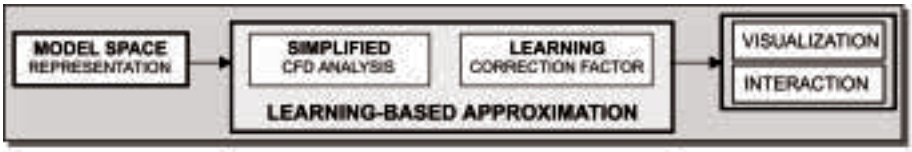

Figure 1: Learning algorithms coupled with simplified fluid field algorithms.

Machine-learning algorithms have been widely used for a variety of applications such as speech recognition, genetic mapping, etc. (Russel and Norvig, 2003). They include both supervised and unsupervised learning approaches. In supervised learning, the agent attempts to generalize the results to obtain the characteristics of the input-output pairs to enable it to choose the optimal actions for those states it has not encountered before. In unsupervised learning, the agent chooses action in the state-space that entails a reward for each action it chooses by evaluating its experience.

Recently, a project was conducted that used a supervised Artificial Neural Network (ANN) as a learning algorithm to predict building thermal performance data (Srinivasan and Malkawi, 2004). A Back Propagation Network (BPN) was used to learn and approximate thermal performance data for rapid visualization purposes. Training datasets for learning were generated by performing CFD simulations. The predicted results were compared with linear-regression datasets for the same interior room. The project demonstrated that while the linear-regression algorithm generated satisfactory results for input ranges that were inside the training range, ANN learning algorithm generated satisfactory results for input ranges that were both inside and outside the training range. Besides, the project also revealed prediction of erroneous thermal performance data if the room shape or volume was altered. This is due to insufficient generalization of the state-action pairs of the ANN model that leads to selection of non-optimal actions. Therefore, for each new spatial configuration, “true” state-action pairs were required for accurate prediction. In other words, massive training datasets were required to predict thermal performance data for a variety of spatial configurations.

Due to infinite possibilities of indoor spatial designs, it is not feasible to obtain training datasets for all possible configurations. This calls for efficient algorithms that adapt to “on-line” learning. One such learning is reinforcement learning that entails the agent choosing an action and obtaining a reward, without relying on supervision (unsupervised learning). RL utilizes a formal framework defining the

interaction between a learning agent and its environment that is aimed at automated goal-directed learning and decision-making (Sutton and Barto, 1998), as shown in Figure 2.

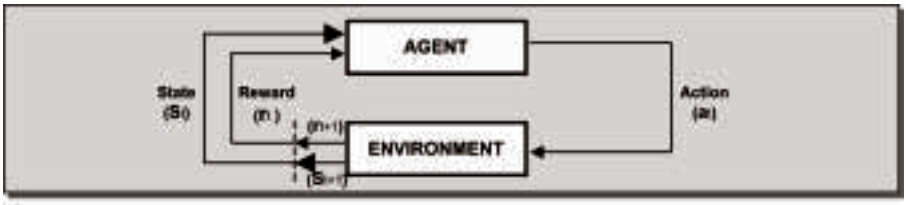

Figure 2: The agent-environment interaction in RL.

The RL approach has been used for a variety of applications such as in elevator dispatching (Crites and Barto, 1995), dynamic channel assignment in cellular phones (Singh and Bertsekas, 1997), learning walking gaits in a legged robot (Huber and Grupen, 2000), etc. This paper discusses this approach and its potential application to bypass intensive thermal simulations in buildings.

2. Learning Methodology

The learning-based estimation involves three components: space modelling, simulation, and learning, as visible in Figure 3.

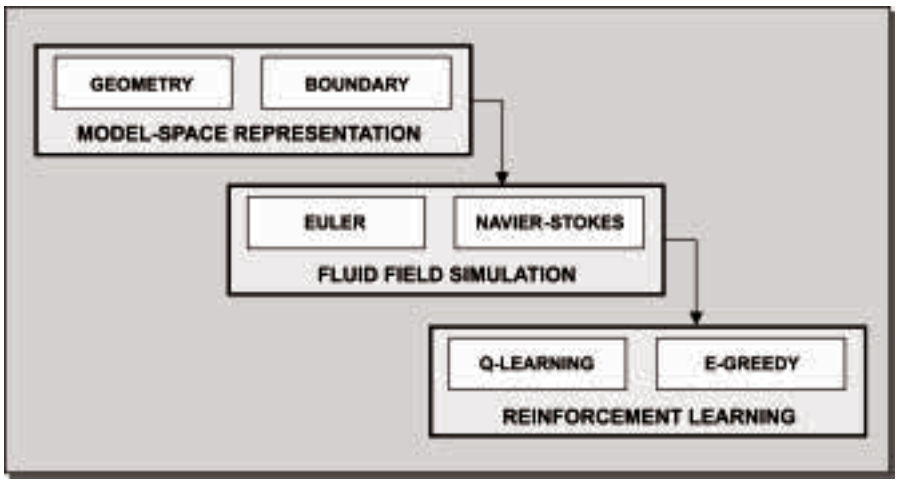

Figure 3: Learning-based estimation method.

The space used for initial training is a rectangular room with dimensions 23.5ft (length) x 15.75ft (width) x 12.00ft (height). The space is located in an interior

zone, i.e. with no exposure to exterior conditions. For simulation purposes, the space is subdivided into smaller grids. In all there are 1253 nodes that represent the indoor space in terms of X-Y-Z axes, (see Figure 4. Six varied configurations were developed by modifying the location of the air supply inlet and the door, Figure 4).

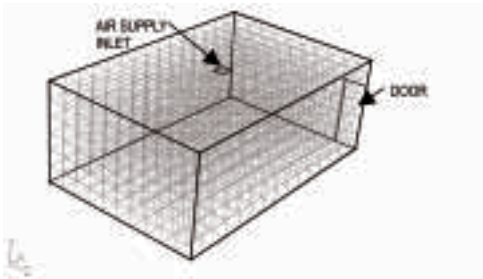

Figure 4: Model-space representation.

Once the space is modelled along with the boundary conditions, it is fed to the simulation engine. While both *Euler* and *Navier-Stokes* CFD algorithms were utilized for predicting the thermal performance data, the convergence of *Euler* algorithm was faster as compared to the *Navier-Stokes* algorithm due to its simplified nature. The *Euler* algorithm offers a simplified system of integral conservation equations for mass, momentum, and energy, and can be written in vector notation as the sum of an area and line integral,

$$\frac{\partial}{\partial t}\iint_{\text{Cell}} \vec{Q}\,dA + \oint_{\text{Cell}} \vec{F}\,dS = 0 \tag{1}$$

$\vec{Q}$ corresponds to the flow, and $\vec{F}$, the flux.

The steady-state solution to the flow field is obtained by marching in time from an initial condition of uniform flow. During the training phase, both algorithms were executed to generate temperature profiles, E_T and N_T respectively. With the knowledge of its location inside the model-space, as well as, the inlet temperature, the RL agent chooses an optimal correction factor (C_{RL}) for *Euler* estimation such that the resultant data (E_{Tnew}) is comparable to *Navier-Stokes* CFD datasets (N_T). In addition, a variable reward function was added for supplying rewards to the agent; the function is dependent on the Mean Square Error (MSE) between the resultant data and Navier-Stokes data.

$$E_T \times C_{RL} = E_{Tnew} \tag{2}$$

$$E_{Tnew} \equiv N_T \tag{3}$$

$$(E_{Tnew} - N_T)^2 = MSE \tag{4}$$

E_T - Temperature estimated by *Euler* algorithm

E_{Tnew} – Resultant temperature

C_{RL} - Correction factor chosen by learning agent

N_T – Temperature predicted by Navier-Stokes algorithm

MSE – Mean Square Error

Reinforcement learning used for this project consists of a *policy* (experience of choosing an action), a variable *reward* function, a *value* function, and the *model* of the environment (state-spaces), as shown in Table 1. While the action space consists of probable actions, i.e. the correction factors to incrementally increase or decrease *Euler* simulated temperature, the state-space consists of the agent's location with respect to the air supply inlet. Depending on the squared error in the deviation of Euler temperature estimation from the regular Navier-Stokes, the agent receives a reward. The reward values are +50 for MSE less than or equal to 1, +100 if MSE is zero, and –(2*MSE) if MSE is greater than 1. The agent's sole objective is to maximize the total reward.

TABLE 1: Reinforcement learning agent architecture.

	Description	*Criteria*
State space	• Air supply inlet, door • [state_space length = 4]	• {x,y,z} position from air supply inlet • Door (outlet) location
Action space	• Increase or decrease (C_{RL}) estimated E_T • [action_space length = 8]	• Increase E_T by {10%, 20%, 30%, 40%} • Decrease E_T by {10%, 20%, 30%, 40%}

Variable reward function	• Mean Square Error (MSE)	• (E_T x C_{RL}) = E_{Tnew}
		• $(E_{Tnew} - N_T)^2$ = MSE
		• if (MSE <=1){reward = +50}
		if (MSE = 0) {reward = +100}
		if (MSE > 1) {reward = -(2*MSE)}

Learning	• Action-value (Q-learning)	• ε-Greedy to explore actions

One of the most accepted and effective RL algorithms is the Q-learning technique. In Q-learning, the agent learns an *action-value* function by supplying the expected utility of taking a given action in a given state, simultaneously collecting experiences (Watkins, 1989; Watkins and Dayan, 1992). The Q-learning technique has been followed in this project,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \gamma \max_{\alpha} Q(s_{t+1}, a_t) - Q(s_t, a_t)] \quad (5)$$

$Q(s_t, a_t)$- State value of taking action (a_t) in state (s_t)

α - Learning rate

γ- Discount rate

r_{t+1} - Reward of taking action (a_t)

The Q-learning parameters include the learning rate ($\alpha = 1.0$), and discounting factor ($\gamma = 0.1$). While learning rate allows the inclusion of new observations into the generalization process, discount rate controls the possible future rewards. Since chosen actions can affect successive rewards, the agent faces a challenge between *exploration* and *exploitation*. In other words, the agent must exploit what it already knows, yet explore to refine its understanding of the rewarding actions available. In this project, a ε-*Greedy* algorithm, a hybrid of a *Blind Search* and a *Greedy* algorithm, was used with the Q-learning algorithm to aid in the selection of best actions that would maximize its total reward.

3. Simulation and Results

Convergence to optimal action is feasible only if the agent visits all possible discrete states and executes each possible action in those states several times. Yet, a near-optimal action to determine the correction factor is attainable via a balanced exploration-exploitation scheme. In this project, convergence is achieved when the agent experiences all the data points. Preliminary results demonstrate the accuracy of estimating the correction factor such that the resultant temperature is comparable to the CFD simulations that utilize *Navier-Stokes* equations. When the graph of the absolute total rewards was plotted against the episodes, it appears that the total rewards had come reasonably close to convergence (Figure 5). The roughness of the plot demonstrates the exploration aspect of Q-learning, which continues to experiment with different actions in search for better estimation. In Figure 6 we see the Mean Square Error values plotted against the episodes.

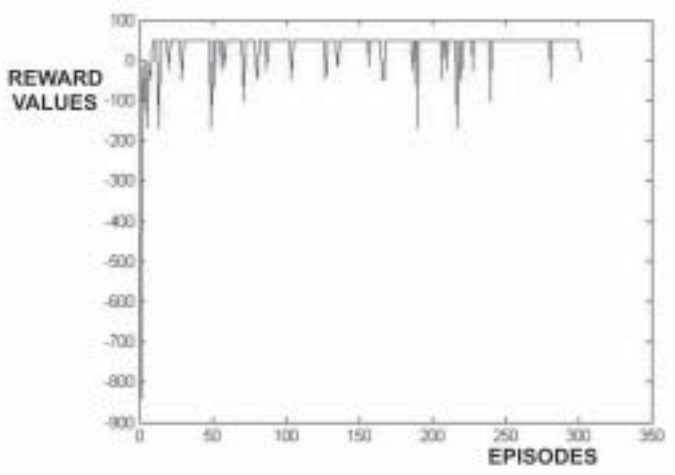

Figure 5: Total rewards Vs episodes.

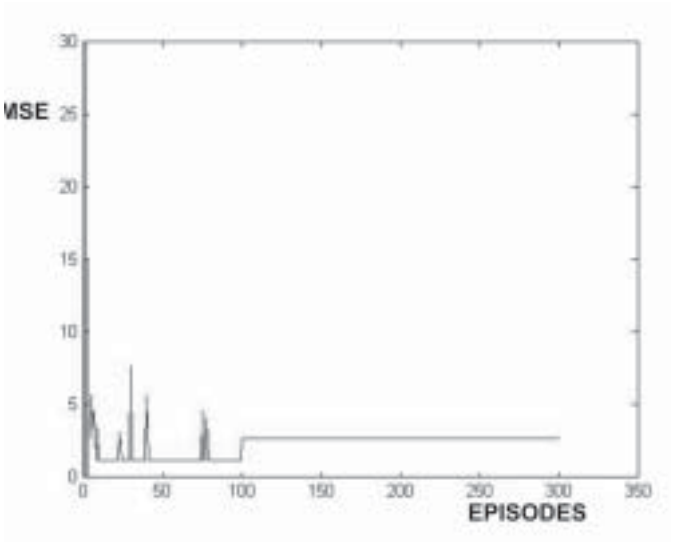

Figure 6: Mean Square Error Vs episodes.

4. Conclusions

Although the use of reinforcement learning was demonstrated to enhance the results of fast fluid algorithms, the project did not investigate the RL potential in maintaining prediction accuracy as the environment changes. The convergence rate of Q-learning was dependent on the number of discrete states. Q-learning using discrete states has an inherent problem, arising from what is known as the "*curse of dimensionality*". In other words, as the number of state variables increases, the number of discrete states increases exponentially, escalating the convergence time. Future work will focus on the selection of decisive learning parameters that would allow rapid convergence to optimality (or near-optimality) for real-time immersive data visualization applications.

References

Bajaj, C.L., Pascucci, V., Thompson, D., & Zhang, X.Y. 1999, Parallel accelerated iso-contouring for out-of-core visualization, *Paralle Visualization and Graphics Symposium.*

Carrilho-da-Graca, G., Chen, Q., Glicksman, L.R., & Norford, L.K. 2002, Simulation of wind-driven ventilative cooling systems for an apartment building in Beijing and Shanghai, *Energy and Buildings*, vol. 34, no. 1, pp. 1–11.

Crites, R.H., & Barto, A.G. 1996, Improving elevator performance using reinforcement learning, *Advances in Neural Information Processing Systems*, vol. 8.

Eckhardt, B. & Zori, L. 2002, Computer simulation helps keep down costs for NASA's "lifeboat" for the international space station, *Aircraft Engineering and Aerospace Technology: An International Journal*, vol. 74, no. 5, pp. 442–446.

Huang, H. & Haghighat, F. 2002, Modeling of volatile organic compounds emission from dry building materials, *Building and Environment*, vol. 37, no. 12, pp. 1349–1360.

Huber, M. & Grupen, R.A. 2000, A hybrid architecture for learning robot control tasks, *Robotics Today*, vol. 13, no. 4.

Lane, D.A. 1995, UFAT—Parallelizing a particle tracer for flow visualization, *SIAM Conference on Parallel Processing for Scientific Visualization.*

Malkawi, A.M. & Srinivasan, R.S. 2005, A new paradigm for human building interaction: The use of CFD and augmented reality, *Automation in Construction Journal*, vol. 14, no. 1, pp. 71–84.

Russel, S. & Norvig, P. 2003, Artificial Intelligence: A modern approach, Prentice Hall, New Jersey.

Singh, S., & Bertsekas, D. 1997, Reinforcement learning for dynamic channel allocation in cellular telephone systems, *Advances in Neural Information Processing Systems,* vol. 10.

Srinivasan, R.S. & Malkawi, A.M. 2004, The use of learning algorithms for real-time immersive data visualization in buildings, *SIGRADI 2004.*

Sutton, R.S. & Barto, A.G. 1998, Reinforcement Learning: An Introduction, MIT Press, Massachusetts.

Ueng, S., Sikorski, C. and Ma, K. 1997, Out-of-core streamline visualization on large unstructured meshes, *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 4, pp. 370–380.

Wasfy, T.M. and Noor, A.K.: 2003, Integrated virtual reality toolkit for effective visualization of CFD datasets in virtual environments, *Proceedings of DETC03*.

Watkins, C.J.C.H. 1989, Learning from delayed rewards, Ph.D. thesis, King's College, Cambridge, U.K.

Watkins, C.J.C.H. & Dayan, P.D. 1992, Q-Learning, *Machine Learning* vol. 8, no. 3, pp. 279-292.

Yeoh, G.H., Yuen, R.K.K., Lu, W.Z. & Chen, D.H. 2003, On numerical comparison of enclosure fire in a multi-compartment building, *Fire Safety Journal*, vol. 38, no. 1, pp. 85–94.