Real-time Simulations Using Learning Algorithms for Immersive Data Visualization in Buildings

Ravi S. Srinivasan and Ali M. Malkawi

IJAC 3-3_final 17/1/06 9:36 am Page 266

Real-time Simulations Using Learning Algorithms for Immersive Data Visualization in Buildings

Ravi S. Srinivasan and Ali M. Malkawi

Computational Fluid Dynamic (CFD) simulations are used to predict indoor thermal environments and assess their response to specific internal/external conditions. Although computing power has increased exponentially in the past decade, CFD simulations are still time-consuming and their prediction results cannot be used for real-time immersive visualization in buildings. A method that can bypass the time-consuming simulations and generate "acceptable" results will allow such visualization to be constructed.

This paper discusses a project that utilizes a supervised Artificial Neural Network (ANN) as a learning algorithm to predict post-processed CFD data to ensure rapid data visualization. To develop a generic learning model for a wide range of spatial configurations, this paper presents a pilot project that utilizes an unsupervised Reinforcement Learning (RL) algorithm. The ANN technique was integrated with an interactive, immersive Augmented Reality (AR) system to interact with and visualize CFD results in buildings. ANN was also evaluated against a linear regression model. Both models were tested and validated with datasets to determine their degree of accuracy. Initial tests, conducted to evaluate the user's experience of the system, indicated satisfactory results.

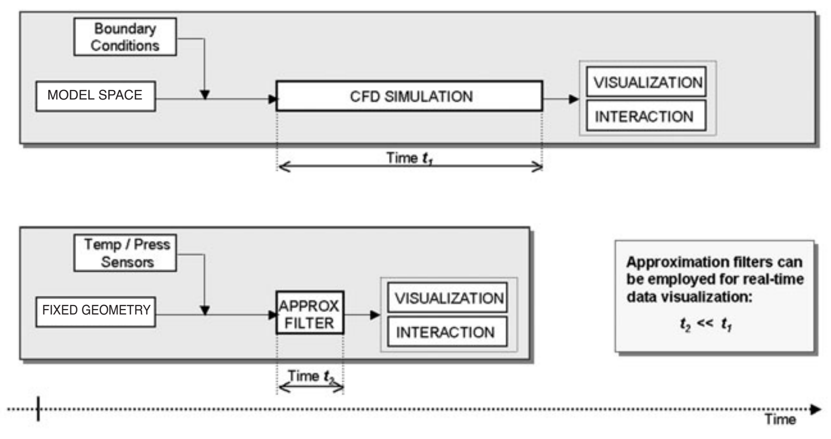
IJAC 3-3_final 17/1/06 9:36 am Page 267

1. Introduction

Building performance analysis allows the architect / engineer to understand the behavior of buildings such as thermal and ventilation performance criteria. While thermal simulation predicts the energy mass flow, ventilation simulation predicts airflow within the design spaces and between them. Such simulations are currently employed in almost all stages of the design, construction, and management of buildings. One such simulation is Computational Fluid Dynamics that utilizes numerical techniques to solve the Navier-Stokes conservation equations of mass, momentum, and thermal energy for fluid fields. CFD simulations of flow movements provide accurate views of three-dimensional behavior of complex systems. CFD simulations have currently been used for building performance analysis including natural ventilation design [1], prediction of smoke and fire [2], building material emissions for Indoor Air Quality (IAQ) assessment [3], and indoor environmental analysis [4]. Visualization of CFD results is vital for accurate understanding by the users. With the advent of new hardware technologies, CFD visualization has transformed from static two-dimensional forms to dynamic three-dimensional modes. Recently, several research efforts were conducted to visualize CFD data using Virtual Environments (VE) such as Virtual Reality (VR) and Augmented Reality environments [5-7]. VR provides total immersion to the user while AR allows partial immersion by overlaying synthetic data objects in real environments. In addition, Human-Computer Interaction (HCI) techniques have been used to interface with CFD data results to allow efficient data manipulation and comprehension [8]. Current technologies permit the visualization of post-processed CFD data. The time delay associated with complex CFD simulation does not allow real-time data visualization. Real-time data visualization of CFD data will enable real-time monitoring of indoor environments. Integration of such real-time visualization of indoor thermal environments with multi-modal HCI will dramatically enhance the way buildings are experienced, managed, and operated.

Despite the 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 simulation equations. Several research projects have been conducted to enable rapid explorations of post-processed data. Examples of such research include the following studies: parallel computation [9], out-of-core particle tracing [10], and load balancing [11]. In addition, techniques utilized to lower the time spent in data generation have also been developed. These include the following: selective visualization [12] and feature extraction [13,14].

Despite the efforts of these researchers, these techniques cannot be employed for real-time data visualization. One approach to solve this problem is using approximation techniques to bypass time-consuming simulations and generate "acceptable" results to allow real-time visualizations, Figure 1. Owing to their generalization, learning algorithms can aid in data approximation.



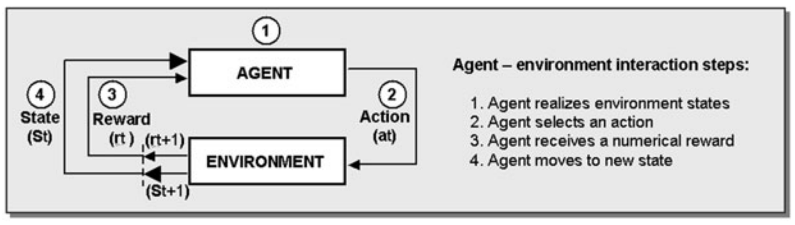
◀ Figure 1: Data generation methods – traditional CFD simulations and data approximation.

Learning algorithms gain knowledge based on the input/output pair of system behavior. These algorithms can be broadly classified as supervised and unsupervised learning, and have been widely used for a variety of applications such as speech recognition, genetic mapping, etc. [15]. In supervised learning, the algorithm is provided with correct output of the function for supplied input value. At every iteration, the algorithm attempts to modify its functional representation to match the output through positive feedback or penalty. Once the input/output function is learnt and added to prior knowledge, the learning algorithm can be employed to predict correct (or approximately correct) output values for given input values. Examples of such learning include linear regression, neural network, etc. On the other hand, unsupervised learning algorithms neither rely on supervision nor require extracting features. In unsupervised learning, the algorithm maps situations to actions while maximizing reinforcement (or rewards), and, therefore, does not need training datasets. Such learning can adapt to online learning methodology and can be utilized for learning without supervision [16]. Examples of unsupervised learning include reinforcement learning, Q-learning, etc. These learning algorithms can be used to develop systems that will predict building behavior based on their learning experience.

This paper describes the use of learning algorithms for real-time data visualization. The paper compares the use of linear regression algorithms with ANN. To develop a generic learning model for a wide range of spatial configurations, the article discusses the use of an unsupervised RL algorithm. In addition to the details of learning algorithms utilized for real-time data visualization, the descriptions of the experimental setup, constraints, and results are presented in this paper.

2. Data Approximation Model

The data approximation model consists of two modules: space representation and learning algorithms, Figure 2. Space representation accounts for room geometry, its properties, and internal obstructions. The learning algorithms are trained using input/output data based on the building thermal behavior.


1
AGENT
4
State
(St)
3
Reward
(rt)
(rt+1)
2
Action
(at)
ENVIRONMENT
(St+1)
Agent – environment interaction steps:
1. Agent realizes environment states
2. Agent selects an action
3. Agent receives a numerical reward
4. Agent moves to new state

► Figure 2: Data approximation model.

The space used for training was a 7.16m (length) × 4.80m (width) × 3.66m (height) room that is located in an interior zone, i.e., with no exposure to exterior conditions. The boundary conditions of the room were constant except for the air and temperature changes caused by a mechanical system through an air diffuser. The training utilized 1224 nodes that represent the indoor space in terms of X-Y-Z axes. The boundary conditions were the temperature (Ti) and pressure (Pi) of the air supply.

To generate the training set, CFD simulations were performed by changing the boundary conditions, specifically changing the inlet-temperature (Ti) from 25°C to 35°C, and varying the inlet-pressure (Pi) from 0.2 to 0.25 Pascal. The inlet-temperature and pressure were incremented in discrete steps by 0.5°C and 0.05 Pascal respectively. CFD simulations were executed with modified boundary conditions. The input and output data were collected as training datasets for training the two learning algorithms, i.e., the linear regression and ANN algorithms. The output of the nodal temperature and velocity in the three-dimensional space was fed to two learning algorithms to investigate their potential in generating acceptable results for rapid visualization. Table 1 presents the input/output parameters for the learning algorithms.

► Table 1. Description of input/output variables.

Parameters		Input/Output t	Data Type
Inlet temperature	T_i	Input	Numeric
Inlet pressure	P_i	Input	Numeric
Temperature at every node	$T_{o(xyz)}$	Output	Numeric
Velocity magnitude at every node	$V_{o(xyz)}$	Output	Numeric

2.1 Learning Using Statistical Analysis

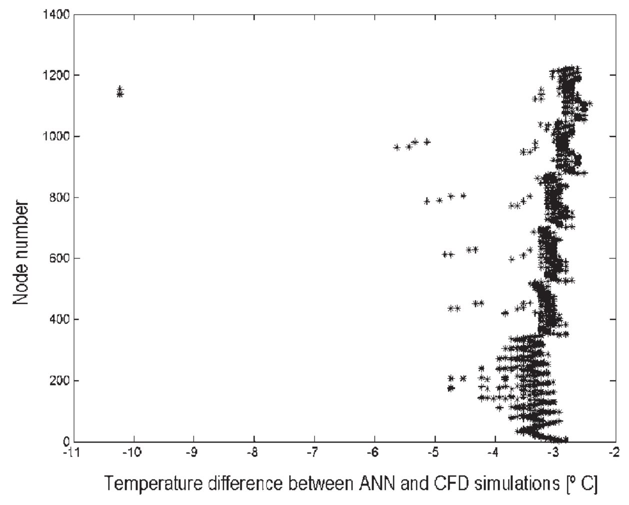
Linear regression is a simple statistical approach for fitting a curve through a set of input/output points with a generalization error [17]. Linear regression allows a variable to correlate with two or more independent variables by *least squared fit*. For comparison purposes, the linear regression algorithm was integrated to the approximation model. Data generated using CFD simulations were used for training. After sufficient training and validation, the linear regression learning algorithm was tested with new inlet temperatures and pressure values, and compared with regular CFD simulations, Figures 3–6.

When the inlet temperature was set within the training range, 30eC in this case, the algorithm generated thermal datasets that only varied between 0.08°C and 0.18°C as compared to regular CFD simulations, Figure 5. This is in sharp contrast when the inlet temperatures were set to temperatures below (10°C and 20°C) and beyond (40°C) the training ranges. The algorithm generated thermal datasets that varied between –5°C and 1.2°C, Figures 3, 4, and 6. During this test phase, a constant pressure was maintained at 0.2 Pascal. Thus, the learning using the linear regression

algorithm was satisfactory only for inlet temperatures that were inside the training range. A different algorithm to allow better generalization is needed. This prompted the use of the ANN learning algorithm described in the next section.

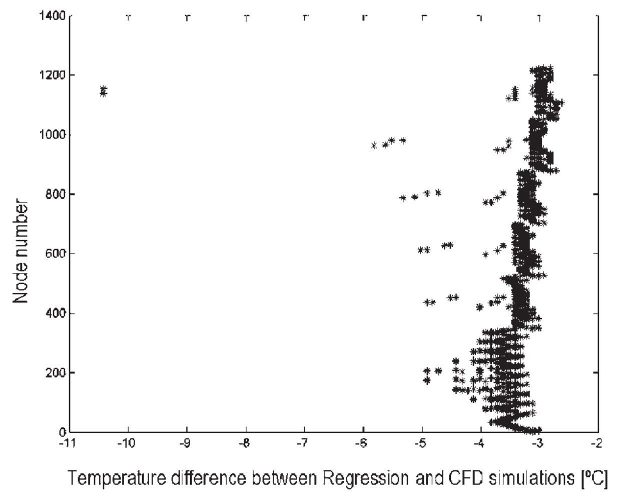
2.2 Learning Using Artificial Neural Network

Back Propagation Network (BPN), a type of ANN, was used. BPN utilizes the *Widrow-Hoff* learning rule that iteratively adjusts the weights of the connection matrix in order to maximize the quality of re-organization of the input parameters [18]. BPNs are the most popular supervised learning method for multi-layer neural networks [19]. The back-propagation generalization allows training a network on a representative set of input and target values and obtaining accurate results without training the network on all possible input/output values. The BPN employed for this project consists

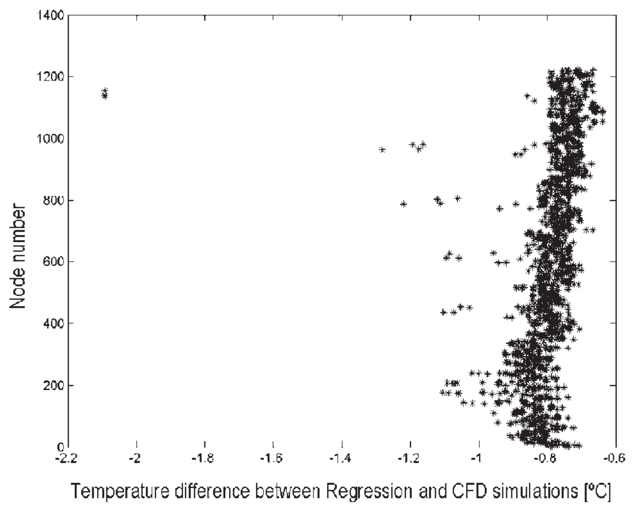


◀ Figure 3. Inlet temperature maintained at 10°C.

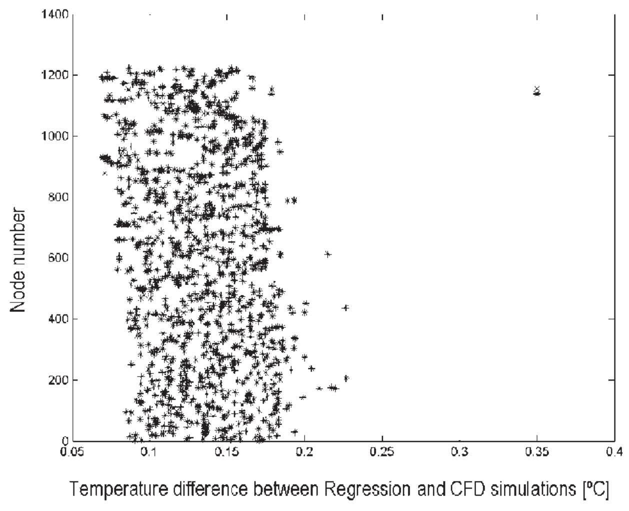
► Figure 4. Inlet temperature maintained at 20°C.



► Figure 5. Inlet temperature maintained at 30°C.

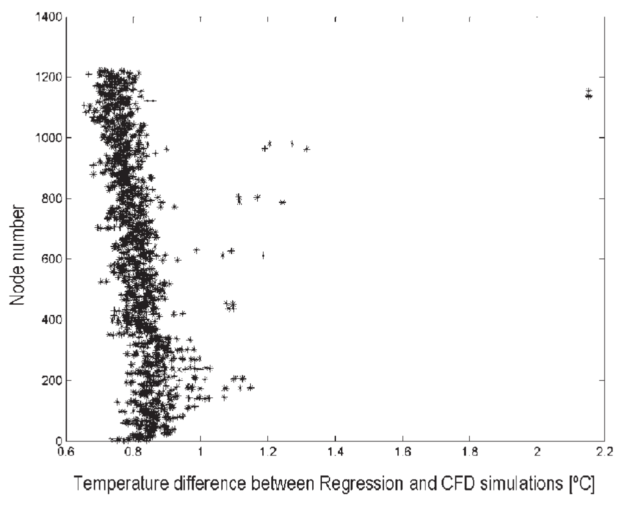


► Figure 6. Inlet temperature maintained at 40°C.

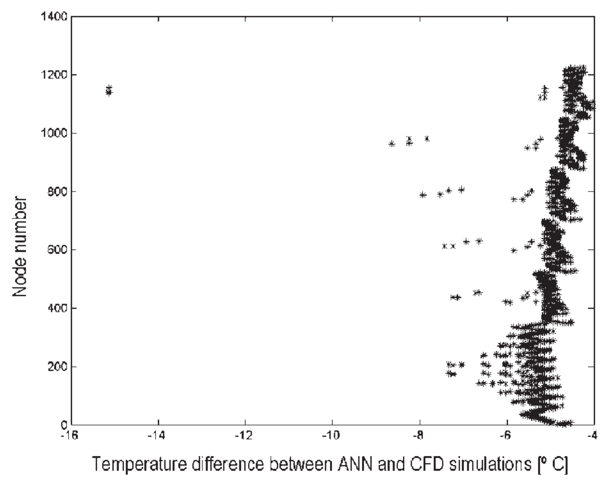


of a four-layered neural network, with two hidden layers and two neurons each that represent the functional relationship between the inputs and outputs. This four-layered sigmoid-linear network represents the functional relationship between inputs and outputs. A *gradient descent* convergence method was selected for this BPN. In addition, different activation Transfer Functions (TF) were applied to hidden layers in order to detect different features during training.

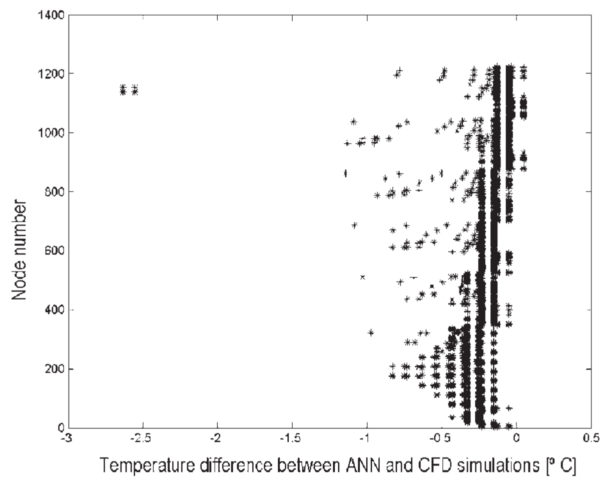
Simulating the ANN learning algorithm with training data resulted in varied weights for input/output variables. The learning algorithm was tested with validation datasets to determine the degree of accuracy the model can achieve. After satisfactory training, the model was provided with new inlet temperature and pressure values to generate corresponding nodal temperature and pressure data. The model converged with a reasonable Mean Square Error (MSE) performance of 0.082. The model predictions were compared with regular CFD simulations, Figures 7–10.



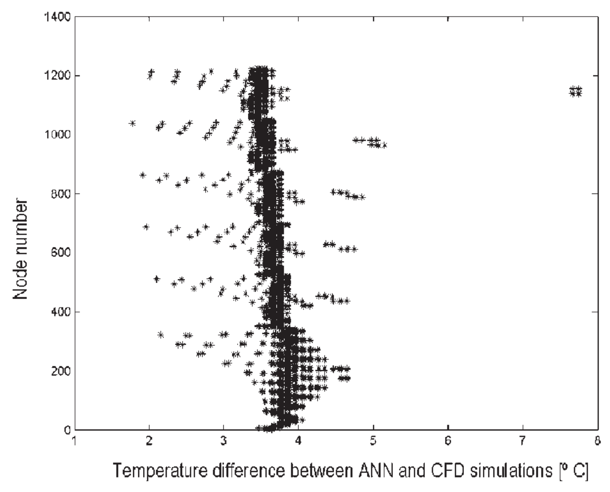
◀ Figure 7. Inlet temperature maintained at 10°C.



◀ Figure 8. Inlet temperature maintained at 20°C.



► Figure 9. Inlet temperature maintained at 30°C.

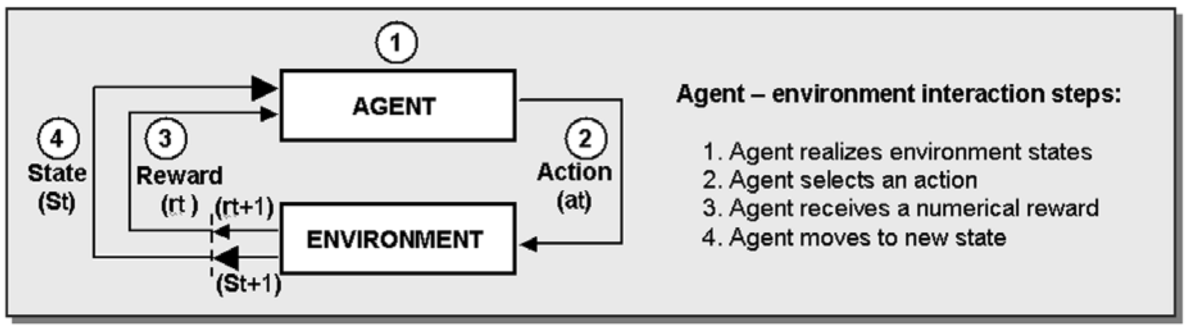


► Figure 10. Inlet temperature maintained at 40°C.

Artificial neural network learning was satisfactory for input ranges that were both inside and outside the training data ranges. However, generalizing with small datasets is difficult using ANN learning algorithms. This might lead to erroneous results. In addition, for any new spatial geometry, new sets of training data are required for sufficient generalization of the algorithm. 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. To develop a generic learning model for real-time data visualization, a more rigorous algorithm is needed. Such an algorithm can generate CFD datasets for a wide range of spatial configurations. This calls for algorithms that adapt to "on-line" learning without relying on supervision. One such learning model is reinforcement learning that utilizes a formal framework defining the interaction between a learning agent and its environment.

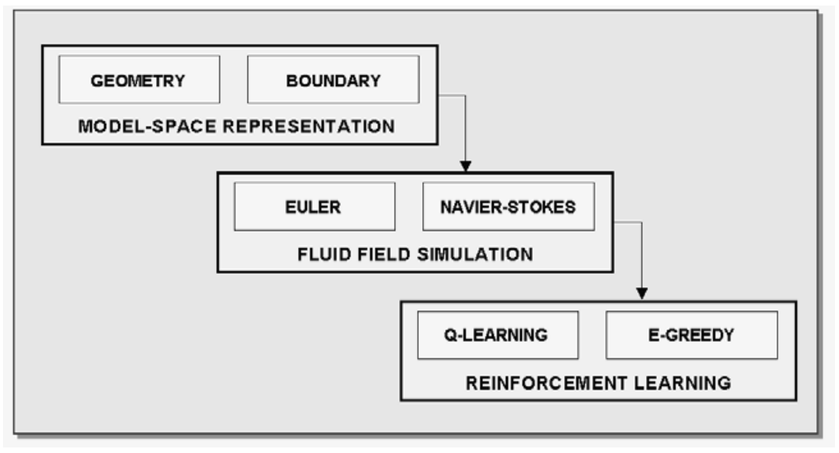
2.3 Learning using Reinforcement Learning

A reinforcement learning agent interacts with the environment by choosing an *action* in the state-space that entails a *reward* for such action by evaluating its previous experience, Figure 11. In this type of learning, the agent must predict the action (through exploration) that will generate maximum payoff in the long run. The RL approach has been used for a variety of applications such as in elevator dispatching [20], dynamic channel assignment in cellular phones [21], learning walking gaits in a legged robot [22], etc. This paper discusses this approach and its potential application to bypass intensive thermal simulations in buildings.

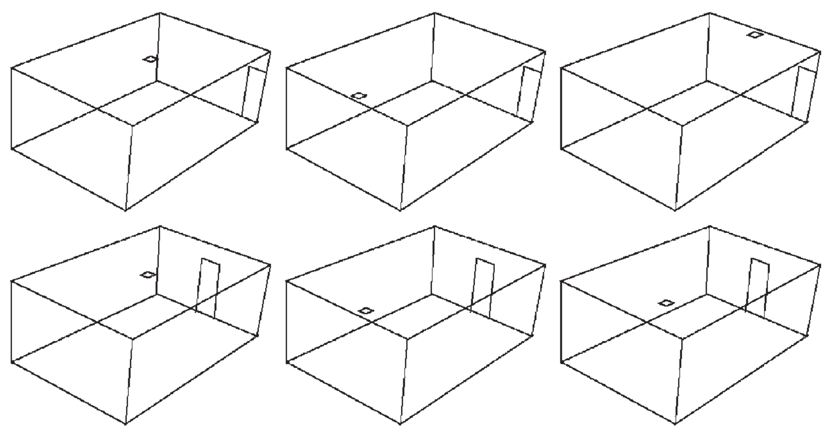

1
AGENT
2
Action
(at)
3
Reward
(rt)
(rt+1)
4
State
(St)
ENVIRONMENT
(St+1)
Agent – environment interaction steps:
1. Agent realizes environment states
2. Agent selects an action
3. Agent receives a numerical reward
4. Agent moves to new state

▲ Figure 11. The RL agent-environment interaction

The learning-based estimation involves three components: space modeling, simulation, and learning, Figure 12. Six varied configurations were developed by modifying the location of the air supply inlet and the door, Figure 13.



◀ Figure 12. Reinforcement learning-based simulation model


▶ Figure 13. ISix configurations by varying the air supply inlet and door position

Once the space is modeled 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. 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. In addition, a variable reward function was added for supplying rewards to the agent; the function is dependent on the MSE between the resultant data and Navier-Stokes data.

$$E_T \times C_{RL} = E_{Tnew} \tag{1}$$

$$E_{Tnew} \equiv N_T \tag{2}$$

$$(E_{Tnew} - N_T)^2 = MSE \tag{3}$$

E_T	– Temperature estimated by Euler algorithm
C_{RL}	– Correction factor chosen by learning agent
E_{Tnew}	– Resultant temperature
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). 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.

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 [23, 24]. The Q-learning technique has been followed in this project,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{\alpha} Q(s_{t+1}, a_t) - Q(s_t, a_t)] \qquad (4)$$

$Q(s_t, a_t)$ – State value of taking action (at) in state (st)

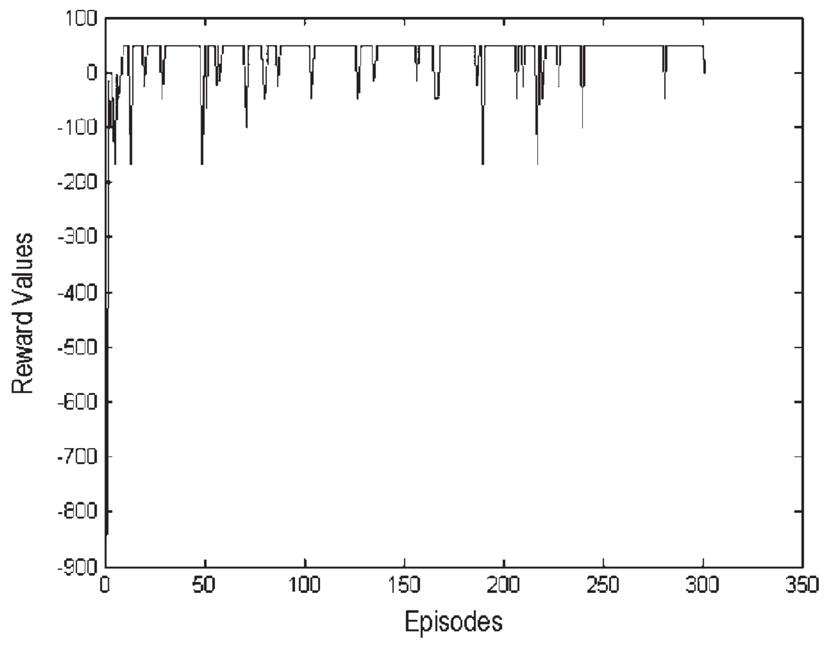
α – Learning rate

γ – Discount rate

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

The Q-learning parameters include the learning rate ($\alpha = 1.0$), and the 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, an *Epsilon-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.

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. Figure 14 shows that the total rewards are reasonably close to convergence. The roughness of the plot demonstrates the exploration aspect of Q-learning, which continues to experiment with different actions in search for better estimation.



▶ Figure 14. Total rewards plotted against episodes

The applicability of the data approximation model was tested using ANN as an example. The model was integrated with an interactive, immersive AR system for real-time data visualization. This system is an interactive speech and gesture recognition-based, immersive AR system specifically designed to visualize and interact with buildings and their thermal environments. Detailed information of the interactive *Human-Building Interaction* system can be found in [25, 26].

3. Conclusions

The data approximation model demonstrated the potential of real-time data visualization with the aid of learning algorithms. Although linear regression and ANN learning algorithms enabled such visualization, the space was limited in terms of geometry and shape. In addition, good generalization of the algorithms requires massive training datasets which are time consuming and often, infeasible. The study showed ANN learning algorithm as a definite improvement over linear regression learning for a fixed geometry. In both cases, modification of spatial geometry required new training datasets. This demands massive training datasets to predict for a variety of spatial configuration, which is not feasible. Although reinforcement learning demonstrated online learning, 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" [27]. In other words, as the number of state variables increases, the number of discrete states increases exponentially, escalating the convergence time. Although, as described earlier, this paper illustrated the potential applications of learning algorithms for real-time data visualization, further validation work is needed to ensure robustness in performance and user evaluation.

References

1. Carrilho-da-Graca, G., Chen, Q., Glicksman, L.R., and Norford, L.K., Simulation of wind-driven ventilative cooling systems for an apartment building in Beijing and Shanghai, *Energy and Buildings Journal*, 2002, 34(1), 1–11.
2. Yeoh, G.H., Yuen, R.K.K., Lu, W.Z., and Chen, D.H., On numerical comparison of enclosure fire in a multi-compartment building, *Fire Safety Journal*, 2003, 38(1), 85–94.
3. Huang, H. and Haghighat, F., Modeling of volatile organic compounds emission from dry building materials, *Building and Environment Journal,* 2002, 37(12), 1349–1360.
4. Eckhardt, B. and Zori, L.:, Computer simulation helps keep down costs for NASA's "lifeboat" for the international space station, *Aircraft Engineering and Aerospace Technology: An International Journal*, 2002, 74(5), 442-–446.
5. Teylingen, R.V., Ribarsky, W., and van der Mast, C., Virtual Data Visualizer, *IEEE Transactions on Visualization and Computer Graphics*, 1997, 3(1), 65–74.
6. Shahnawaz, V., Vance, J., and Kitti, S., Visualization of Post Processed CFD Data in a Virtual Environment, in: *Design Engineering Technical Conference*, American Society of Mechanical Engineers, 1999, 1–7.
7. Wasfy, T.M., and Noor, A.K., Rule-based Natural-Language Interface for Virtual Environments, *Advances in Engineering Software*, 2002, 33(3), 155–168.
8. Malkawi, A., Immersive Building Simulation, in: Malkawi, A., and Augenbroe, G., eds., *Advanced Building Simulation*, Spon Press, London, 2004, 217–246.
9. Lane, D.A., Parallelizing a Particle Tracer for Flow Visualization, in: Bailey, D.H., Bjorstad, P.E., Gilbert, J.R., Mascagni, M., Schreiber, R.S., and Simon, H.D., eds., *Seventh SIAM Conference on Parallel Processing for Scientific Computing*, SIAM, San Francisco, California, 1995, 784–789.
10. Ueng, S., Sikorski, C., and Ma, K., Out-of-core Streamline Visualization on Large Unstructured Meshes, in: *IEEE Transactions on Visualization and Computer Graphics*, 1997, 3(4), 370-380.
11. Bajaj, C.L., Pascucci, V., Thompson, D., Zhang, X.Y., Parallel Accelerated Isocontouring for Out-of-core Visualization, in: *IEEE Symposium on Parallel Visualization and Graphics*, IEEE, San Francisco, California, 1999, 97–104.
12. Walsum, T., *Selective Visualization on Curvilinear Grids*, PhD Thesis, Delft University of Technology, Delft, The Netherlands, 1995.
13. Walsum, T., Post, F.H., and Silver, D., Feature Extraction and Iconic Visualization, in: *IEEE Transactions on Visualization and Computer Graphics*, 1996, 2(2), 111–119.
14. Kenwright, D., Henze, C., and Levit, C., Feature Extraction of Separation and Attachment Lines, in: *IEEE Transactions on Visualization and Computer Graphics*, 1999, 5(2), 135–144.
15. Russel, S., and Norvig, P., *Artificial Intelligence: A Modern Approach*, Prentice Hall, New Jersey, 2003.

16. Sutton, R.S., and Barto, A.G., *Reinforcement Learning: An Introduction*, MIT Press, Masssachusetts, 1998.

17. Draper, N.R., and Smith, H., *Applied Regression Analysis*, New York: Wiley & Sons, 1966.

18. Widrow, B., and Hoff, M.E., Adaptive Switching Circuits, in: *IRE WESCON Convention Record*, 1960, 96–104.

19. Rumerlhart, D.E., and McClelland, J.L., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Cambridge, Massachusetts, MIT Press, 1986.

20. Crites, R.H., and Barto, A.G., Improving elevator performance using reinforcement learning, in: Touretzky, D.S., Mozer, M.C., and Hasselmo, M.E., eds., *Advances in Neural Information Processing Systems*, 1996, 8, 1017–1023.

21. Singh, S., and Bertsekas, D.: 1997, Reinforcement learning for dynamic channel allocation in cellular telephone systems, in: Mozer, M.C., Jordan, M.I., and Petsche, T., eds., *Advances in Neural Information Processing Systems*, 9, 974–980.

22. Huber, M., and Grupen, R.A., A hybrid architecture for learning robot control tasks, *Robotics Today*, 2000, 13(4).

23. Watkins, C.J.C.H., *Learning from delayed rewards*, PhD Thesis, King's College, Cambridge, U.K, 1989.

24. Watkins, C.J.C.H., and Dayan, P.D., Q-Learning, *Machine Learning*, 1992, 8(3), 279–292.

25. Malkawi, A., and Srinivasan, R., A New Paradigm for Human-Building Interaction: The Use of CFD and Augmented Reality, *Automation in Construction Journal*, 2005, 14(1), 71–84.

26. Malkawi, A., and Srinivasan, R., Interfacing with the Real World and its Performance, *International Journal of Architectural Computing*, 2005, 3(1), 43–56.

27. Bellman, R., *Adaptive Control Processes: A Guided Tour*, Princeton University Press, Princeton, New Jersey, 1961.

Ravi S. Srinivasan and Ali M. Malkawai
University of Pennsylvania
Department of Architecture
210 S. 34th Street,
Philadelphia PA 19104, USA

{sravi, malkawi}@design.upenn.edu