

Santa Clara University

Scholar Commons

Engineering Ph.D. Theses

Student Scholarship

9-2020

Cost effective and Non-intrusive occupancy detection in residential building through machine learning algorithm

Chenli Wang

Follow this and additional works at: https://scholarcommons.scu.edu/eng_phd_theses



Part of the [Mechanical Engineering Commons](#)

Santa Clara University

Department of Mechanical Engineering

Date: Sep, 2020

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY
SUPERVISION BY

Chenli Wang

ENTITLED

**COST EFFECTIVE AND NON-INTRUSIVE OCCUPANCY DETECTION IN
RESIDENTIAL BUILDING THROUGH MACHINE LEARNING ALGORITHM**

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF

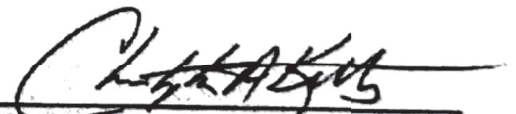
DOCTOR OF PHILOSOPHY IN MECHANICAL ENGINEERING



Thesis Advisor
Dr. Hohyun Lee



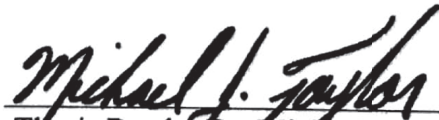
Chair of Department
Dr. Drazen Fabris



Thesis Reader Dr. Christopher Kitts



Thesis Reader Dr. On Shun Pak



Thesis Reader Dr. Michael J. Taylor



Thesis Reader Dr. Yi Fang

**COST EFFECTIVE AND NON-INTRUSIVE OCCUPANCY DETECTION IN
RESIDENTIAL BUILDING THROUGH MACHINE LEARNING
ALGORITHM**

By

Chenli Wang

Thesis

Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Mechanical Engineering
in the School of Engineering at
Santa Clara University, 2020

Santa Clara, California

Abstract

Residential and commercial buildings consume more than 40 % of energy and 76 % of electricity in the U.S. Buildings also emit more than one-third of U.S. greenhouse gas emissions, which is the largest sector. A significant portion of the energy is wasted by unnecessary operations on heating, ventilation, and air conditioning (HVAC) systems, such as overheating/overcooling or operation without occupants. Wasteful behaviors consume twice the amount of energy compared to energy-conscious behaviors. Many commercial buildings utilize a building management system (BMS) and occupancy sensors to better control and monitor the HVAC and lighting system based on occupancy information. However, the complicated installation process of occupancy sensors and their long payback period have prevented consumers from adopting this technology in the residential sector. Hence, I explored a method to detect the presence of an occupant and utilize it to reduce energy wasting in residential buildings.

Existing methods of occupancy detection often focus on directly measure occupancy information from environmental sensors. The validity of such a sensor network highly depends on the room configurations, so the approach is not readily transferrable to other residential buildings. Instead of direct measurement, the proposed scheme detects the change of occupancy in a building. The new scheme implements machine learning methods based on a sequence of human activities that happens in a short period. Since human activities are similar regardless of house floorplan, such an

approach may lead to readily transferrable to other residential buildings. I explored three types of human activity sensor to detect door handle touch, water usage, and motion near the entrance, which are highly correlated with the change of occupancy. The occupancy change is not only based on one single human activity, it also depends on a series of human activities that happen in a short period, called event. As the events have different durations and cannot be readily applicable to existing machine learning models due to varying input matrix sizes. Hence, I devised a fixed format to summarize the event regardless of the total duration of the event. Then I used a machine learning model to identify the occupancy change based on the event data. The saving potential of occupancy driven thermostat is about 20 % of energy in residential buildings. However, the actual saving impact in any given house can vary significantly from the average value due to the large variety of residential buildings. Existing building simulation tools did not readily consider the random nature of occupancy and users' comfort. For this reason, I explored a co-simulation platform that integrates an occupancy simulator, a cooling/heating setpoint control algorithm, a comfort level evaluator, and a building simulator together. I explored the annual energy saving impact of an occupancy-driven thermostat compare with a conventional thermostat. The simulation had been repeated in five U.S. cities (Fairbanks, New York City, San Francisco, Miami, and Phoenix) with distinctive climate zones.

Acknowledgements

I would like to express my gratitude to Dr. Hohyun Lee, my advisor, for his patient, encouragement, and guidance throughout the years at Santa Clara University. I learned a lot from him, which is not just about academics, and I have been very grateful to have him as my advisor during my Ph.D. study.

I would like to give special thanks to my committee members. I owe a debt for gratitude to Dr. On Shun Pak for his time and careful attention to detail. To Dr. Yi Fang, I thank him for his tireless support and guidance on Machine Learning Algorithm and coding skills. To Dr. Christopher Kitts, I appreciate his invaluable feedback on my whitepaper that improved my work. I wish to thank Dr. Michael J. Taylor for his helpful advice on word choosing and technical writing.

I must thank all my friends, classmates and roommates, thank you for listening and offering me advice through this entire process. Special thanks to my labmates: Joseph Singer, Michael Giglio, Tom Watson, Kyle Pietrzyk, Joshua Vincent, Kaleb Pattawi and PJ McCurdy. It is always a pleasure to talk with them in lab and know I am not the only one.

I must also acknowledge Dr. Yuhong Liu and Jun Jiang for their advice on data processing and trust factor. Also, I thank Donald MacCubbin at the Machine Shop who

offered design and build advise for constructing door handle touch sensor employed in this work.

I would like to thank my parents for their unconditional love, encouragement, and believing in me. Without their love and support I would never have finished.

To Rachel Lu, my beloved wife, thank you and love you 3000.

Table of Contents

Abstract.....	ii
Acknowledgements.....	iv
Table of Contents.....	vi
List of Figures	x
List of Tables	xv
Nomenclature	xvii
Abbreviations	xix
Chapter 1. Introduction.....	1
1.1. Review of Cyber-Physical System.....	7
1.2. Review of Occupancy Sensors.....	9
1.3. Review of Occupancy Detection via Machine Learning.....	12
1.4. Two-layer Occupancy Detection System.....	15
1.5. Energy Saving Impact of Occupant Driven Thermostat.....	17
1.6. Thermal Comfort Standard and Adaptive Control.....	20
1.7. Thesis Objectives	23
Chapter 2. Economic Non-Intrusive Human Activity Detection Models	26
2.1. Door Handle Touch Detection	27

2.1.1.	Detection of Temperature Change	28
2.1.2.	Criteria Determination	29
2.1.3.	Uncertainty Control with Data Collection	32
2.1.4.	Experimental Set-up.....	35
2.1.5.	Results & Discussion	38
2.2.	Water Usage Detection	40
2.2.1.	Experimental Set-up.....	42
2.2.2.	Results & Discussion	43
2.3.	Passive Infrared Motion Sensor	45
Chapter 3.	Occupancy Detection via Machine Learning	49
3.1.	HLA Based Simultaneous Data Collection	51
3.2.	Machine Learning Algorithms	56
3.2.1.	Decision Tree	56
3.2.2.	Random Forests.....	57
3.2.3.	K-Nearest Neighbors.....	58
3.2.4.	Support Vector Machine.....	58
3.3.	Experimental Set-up	59
3.4.	Data Processing	61
3.5.	Results & Discussion.....	66
Chapter 4.	Energy Saving Impact of Occupancy Information.....	76

4.1.	Occupancy Simulator	77
4.2.	Building Energy Simulation Tool.....	79
4.3.	Thermal Comfort Standard	81
4.4.	Traditional Thermostat Control	85
4.5.	Adaptive Thermostat Control.....	88
4.6.	Co-Simulation Platform.....	89
4.7.	Results & Discussion	91
Chapter 5.	Conclusion & Future Work.....	100
5.1.	Future Work: Quantification of Occupancy	102
5.2.	Future Work: Personal Preference of Setpoint	102
5.3.	Future Work: Other Human Activity Detection Sensor.....	104
5.4.	Future Work: Commercialization	106
Reference		108
Appendices		122
Appendix A.	Human Activity Detection Code (Matlab).....	122
Appendix B.	Temperature Data Collection (LabView)	129
Appendix C.	PIR Motion Sensor Collection (BBB)	133
Appendix D.	Relative Time Event Sequence Format.....	137
Appendix E.	Machine Learning	144
Appendix F.	Decision Tree	147

Appendix G.	Detailed Result: Without Human Activity Layer	148
Appendix H.	Detailed Result: With Human Activity Layer	152
Appendix I.	Cost Estimation of a Sensor System	156
Appendix J.	TCP/IP Host in UCEF	157
Appendix K.	Data Fusion Code in UCEF	163
Appendix L.	Occupancy Simulator and Thermostat Control.....	174
Appendix M.	Comfort Level Evaluate	183
Appendix N.	EnergyPlus IDF Incorporated IDF Model.....	188
Appendix O.	FMU File (C code)	275
Appendix P.	XML File for FMI	287

List of Figures

Figure 1: Increasing trend of the annual energy consumption in the U.S. normalized to the amount in 1949 (data from EIA ¹). The proportion of building energy consumption to total annually energy consumption is also increasing over the past seventy years.	1
Figure 2: Illustration of the change in global surface temperature relative to 1951-1980 average temperatures (data from NASA ²).	3
Figure 3: Annual household site end - use consumption in the U.S. (data from Energy.gov ¹⁴).....	5
Figure 4: Energy-unaware behavior uses twice as much energy as the minimum that can be achieved (data original from WBCSD ¹⁶).....	6
Figure 5: Overall structure of a Cyber-Physical Systems.....	8
Figure 6: Representation the schematic of data processing from the environmental sensors to the final predicted occupancy information.	17
Figure 7: Distribution of gas savings for existing smart thermostats. (data original from NEST white paper ⁵¹).....	18
Figure 8: Raw temperature profile measured by K-type thermocouple with huge fluctuations, about 0.5 °C, compare with temperature profile after using moving average method which keeps temperature trend, filtering out fluctuation.	29
Figure 9: Flowchart of touch detection method.	34

Figure 10: a) Schematic diagram of experimental set-up measuring temperature of this sensor system with a K-type thermocouple. b) Experimental set-up attached on the top side of the door handle with some two-sided tape. Set-up is connected to a NI Compact DAQ to measure and record the temperature data every 0.1 second.	36
Figure 11: Comparison of theoretical maximum temperature change in 0.1 second and actual temperature and temperature change profile with experimental set-up..	37
Figure 12: Demonstration of detected touch events by using normalized temperature change rate data and threshold.	39
Figure 13: Demonstration of no misdetection during this 3 hours long test. Normalized temperature change rate keeps at the level about 0 when the temperature increased with the ambient temperature. During the process, normalized temperature change rate.....	40
Figure 14: Schematic diagram of experimental set-up measuring temperature of this sensor system with a K-type thermocouple.	43
Figure 15: Demonstration of a single water draw event data in a selected minute, where the normalized temperature change rate increased rapidly and exceeded the threshold during the water usage event.	45
Figure 16: PIR motion sensor is installed on the top of the inside door frame to detect any movement near the door area.	47
Figure 17: Description of two selected cases: (A) grab a cell phone and (B) two users back to home in sequence. Both cases contain two outside door handle touch activities and two door area motion activities.	50

Figure 18: Instead of having individual connectivity between entities, the RTI environment can more simply connect federates for co-simulations.....	52
Figure 19: Flowchart representing how data is collected, processed and sent to the database via TCP/IP.....	54
Figure 20: Schematic diagram of the proposed HLA based simultaneous data collection system. Two PC entities connect via TCP/IP forwarder. Two BeagleBone Black entities remotely communicate with RTI. Database entity is operating on the same computer that runs HLA.	55
Figure 21: Schematic floor plan of experimental set-up detecting occupancy information in a living lab with multiple types of sensors and 5 switches for ground truth collection.....	60
Figure 22: Histogram of event duration of 489 events collected in this work.	62
Figure 23: An example of an entering event window: the event started with an outside door handle touch activity followed by motion near the door area. The data from this event window can be converted into a fixed-length format.	64
Figure 24: Schematic diagram of data processing from the variable-size event window to the final predicted occupancy information.....	65
Figure 25: Comparison of the total detection areas and the number of sensors used by different research efforts in residential homes. All models used in these researches were able to provide valid occupancy information, with an error of misclassification less than 5 %. ^{38,39,41,44,46,99,103,104}	72

Figure 26: Demonstration of the importance of five human activities (door handle touches, motion near the door area, and water usages) in determining the occupancy prediction by Random Forest algorithm.	73
Figure 27: Demonstration of the probability of occupancy in the house at different times of day.	78
Figure 28: Illustration of the flowchart of the occupancy simulator.....	79
Figure 29: Acceptable operative temperature ranges based on mean outdoor temperature defined by adaptive model. The comfortable zone (blue grid area) and acceptable zone (yellow pattern area) depict the temperature ranges accepted by 90 % and 80 % of people accordingly.....	83
Figure 30: Display the flowchart of the comfort level evaluator.....	84
Figure 31: Demonstrate the flowchart of the HVAC system controller based on three different control algorithm and operation conditions.....	86
Figure 32: Illustration of the occupancy-driven control algorithm in a cold day at Fairbanks, AK. The outdoor temperature is displayed in yellow dash line. The HVAC system turns on and off based on the occupancy information (shaded area), and the room temperature (red thin solid line) fluctuates around the setpoint (blue thick solid line) when the room is occupied.....	87
Figure 33: The flowchart of the HVAC controller with adaptive setpoints based on different operation conditions.	89
Figure 34: Representation the schematic of data transfer between building simulator, occupant simulator and HVAC controller.	91

Figure 35: Demonstration of the result of annual comfort result and energy consumption at New York City, NY. The stacked bar chart shows the ratio of comfort zone (blue shaded area), acceptable zone (yellow pattern area) and uncomfortable zone (red blank area). Annual energy usages are displayed with black star marks and y-axis on the right.....	92
Figure 36: Comparison of the annual comfortable ratio and saving impact of the Fixed setpoint - Occupancy, Adaptive - Always On, and Adaptive - Occupancy in five different locations.....	96
Figure 37: Demonstration of a set of temperature measurement data of a door handle with multiple touches from two users.	104

List of Tables

Table 1: Summary of existing occupancy sensors in the market.	12
Table 2: Summary of machine learning algorithms and type of sensors for occupancy measurement.	14
Table 3: List of the key parameters used in door handle touch detection model.	34
Table 4: Results of touch testing compared with the ground truth	38
Table 5: List of the key parameters used in door handle touch detection model.	42
Table 6: Results of Toilet Water Usage and Tap Water Usage events compared with expected values.....	44
Table 8: Comparison of the performance of applying Decision Tree, Random Forest, K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) algorithms to the data with and without the Lower Layer System (Human Activities Layer).....	68
Table 9: Demonstration of the confusion matrix of Random Forest models. The matrix shows there is only one misclassified event in the testing data set.....	70
Table 10: Illustration of the detailed information of the misclassified case.....	71
Table 11: Comparison of the validity of Random Forest Models with and without the water usage human activities.....	74
Table 12: Summarization of the detailed weather features of the chosen cities from five different climate zones.	81

Table 13: Summary of the HVAC system setpoint based on three different control algorithm and operation conditions. P is the probability of occupancy.....	86
Table 14: Comparison of the annually comfortable level, energy saving impact, and energy usage in Fairbanks, New York City, San Francisco, and Miami Phoenix.....	94
Table 15: List of monthly saving and payback period of three different control strategies (Fixed setpoint - Occupancy, Adaptive - Always On, and Adaptive – Occupancy), compared with the most wasteful control strategy (Fixed setpoint - Always On) and energy-conscious control strategy (Fixed setpoint – Schedule Based).	98

Nomenclature

T_{raw}	measured temperature, [°C]
T	moving average temperature, [°C]
dT	temperature change, [°C]
$\frac{dT}{dt}$	temperature change rate, [°C/s]
T_i	initial temperature of the door handle, [°C]
T_∞	ambient temperature, [°C]
T_{finger}	human finger temperature, [°C]
$\Delta\dot{T}$	normalized temperature change rate, [1/s]
$\overline{\Delta\dot{T}}$	average normalized temperature change rate, [1/s]
$\overline{\overline{\Delta\dot{T}}}$	mean of average normalized temperature change rate, [1/s]
$\sigma_{\overline{\Delta\dot{T}}}$	standard deviation of average normalized temperature change rate, [1/s]
z	confidence level coefficient
\dot{Q}	heat transferred to the door handle [W]
m	mass of the door handle [g]
C_p	specific heat capacity, [J/g·K]
M	moving average coefficient
t	time, [s]
Δt	time step, [s]
A	touching area [m ²]

h	overall heat transfer coefficient, [W/m ² ·K]
k_{skin}	thermal conductivity of human skin, [W/m·K]
L_{finger}	half of the thickness of the finger, [m]
T_{W_ground}	temperature of water from the ground, [°C]
p	probability of one data point (object) being categorized into a class, [%]
$(.)_N$	previous N touch events data
$(.)_{N+1}$	previous N touch events and new touch event data

Abbreviations

BMS	Building Management System
HVAC	Heating, Ventilation, and Air Conditioning
EIA	Energy Information Administration
CPS	Cyber-Physical System
PIR	Passive Infrared
EM	Electromagnetic
Wi-Fi	Wireless Fidelity
GPS	Global Positioning System
RFID	Radio Frequency Identification
PPM	Parts-per-Million
ML	Machine Learning
BCTVB	Building Controls Virtual Test Bed

HLA	High Level Architecture
ASHRAE	American Society of Heating, Refrigerating and Air- Conditioning Engineers
PMV	Predicted Mean Vote
RLS	Recursive Least-Squares
RTI	Runtime Infrastructure
KNN	K-Nearest Neighbors
SVM	Support Vector Machine
TP	True Positive
TN	True Negative
FN	False Negative
FP	False Positive
ATUS	American Time Use Survey
DOE	Department of Energy

NIST	National Institute of Standards and Technology
UCEF	Universal CPS Environment for Federation
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit

Chapter 1. Introduction

Over the past 70 years, energy demand in the U.S. has tripled.¹ Among all the end-use sectors, buildings have the highest growth rate, which is 52 % more than the growth rate of total energy consumption.¹ Figure 1 shows that the total annual energy consumption from buildings has grown by four times, from 9268 trillion Btu to 39387 Btu.¹ The contribution of building energy usage to total has raised from 29 % to 40 %.

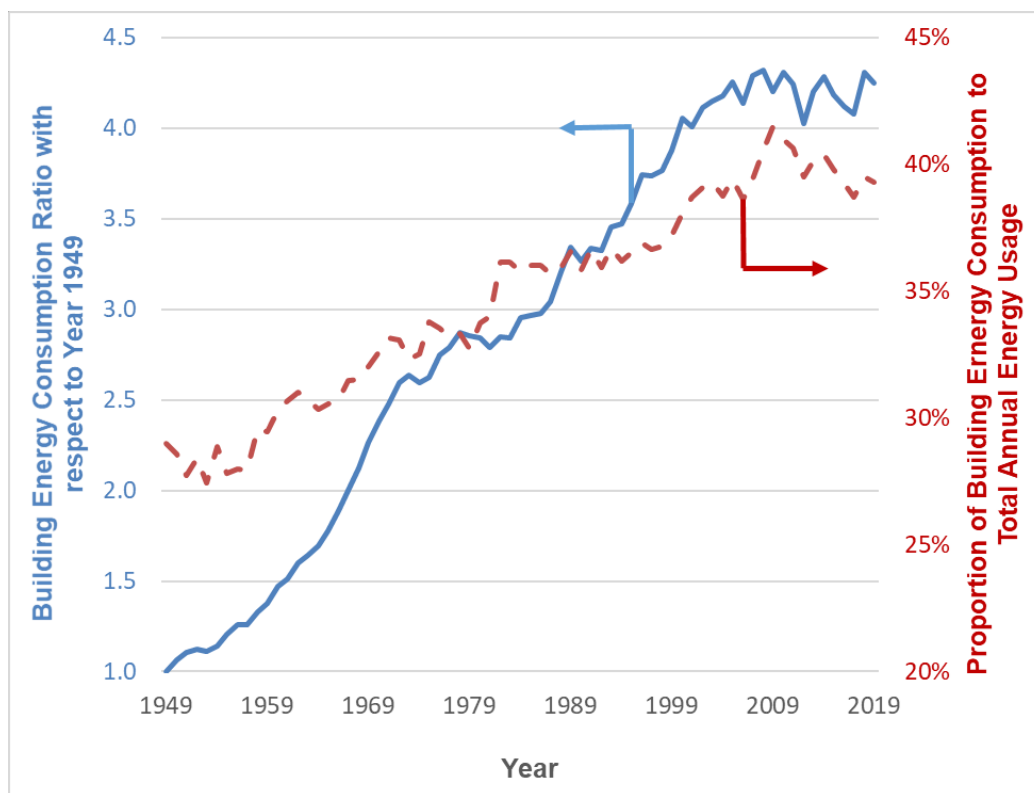


Figure 1: Increasing trend of the annual energy consumption in the U.S. normalized to the amount in 1949 (data from EIA¹). The proportion of building energy consumption to total annually energy consumption is also increasing over the past seventy years.

Increased energy consumption is accompanied by an undesirable impact on our environment and sustainability. Based on the data from NASA,² temperature readings from around the globe during the past century show a relatively rapid increase, which is strongly correlated to the greenhouse gases emissions as people burn fossil fuels. Figure 2 illustrates the change in global surface temperature relative to a long-time average temperature (1951-1980). The average global temperatures are stable before 1960 and consistently increase in the past 50 years. The year 2016 ranks as the warmest year on the record which is 1.02 °C more than the long-time average temperature. A report³ from the Intergovernmental Panel on Climate Change predicted that the degree of the temperature anomaly is likely to reach 1.5 °C around 2030. The average sea level has been rising 3 millimeters per year during the past 30 years,⁴ threatening the areas with low elevation. For example, Bangladesh, one of the most densely populated countries in the world, is estimated to lose 18 % of its land if sea level rises about 1 meter, affecting millions of people. On the global scale, 1 meter rising will take over 220,703 km² country area and impact 57 million people.⁴

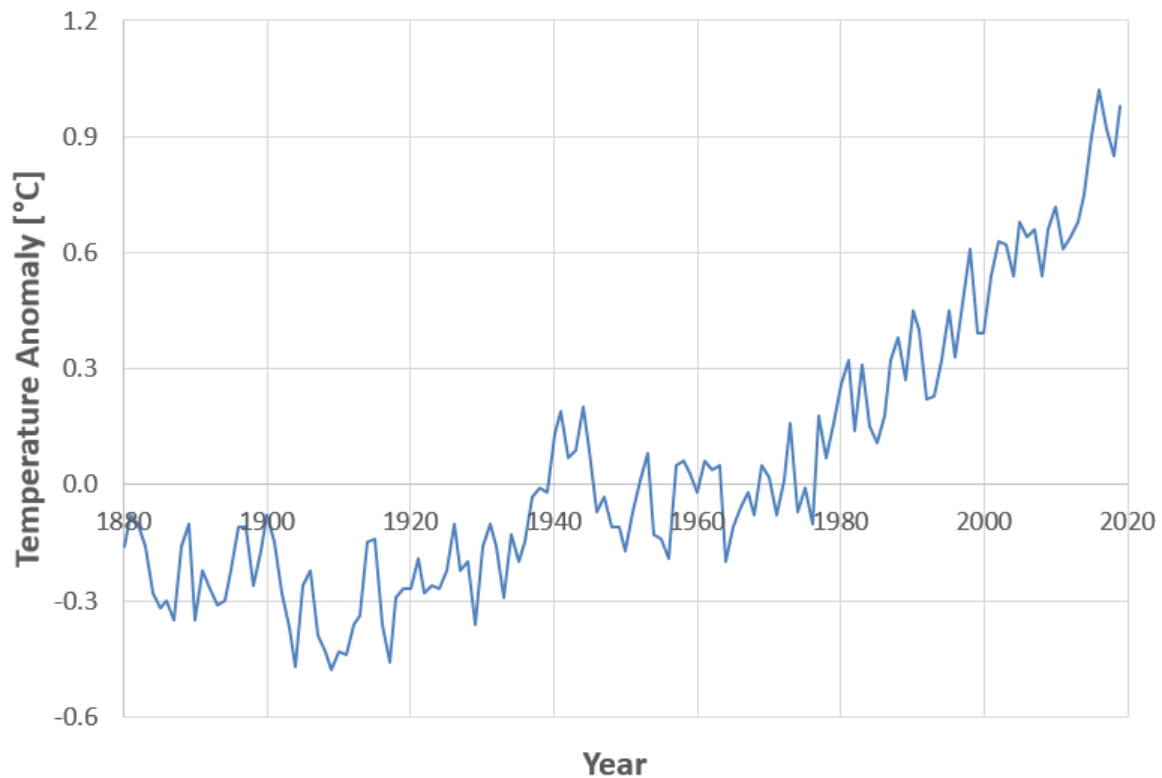


Figure 2: Illustration of the change in global surface temperature relative to 1951-1980 average temperatures (data from NASA²).

The power generation from renewable energy sources can slow down the threat on the future sustainability. However, due to the high cost and uncertainty in output, the adoption of such renewable energy resources in buildings is limited. For example, the output of solar panels is highly dependent on sunlight. Therefore, cloudy, rainy days can have a noticeable effect on the solar energy system. The typical solar payback period in the U.S. is about 8 years.⁵ Moreover, solar panels require a lot of space and some roofs are not big enough to fit the number of solar panels. According to an EIA report in 2019, only 2.8 % of total building energy consumption (1105 trillion Btu) is coming from renewable energy sources.⁶

Another approach for reducing fossil fuel reliance is to use more efficient devices or appliances. Since 1992, ENERGY STAR products saved more than 4 trillion kilowatt-hours of electricity and reduced over 3.5 billion metric tons of greenhouse gas, equivalent to the annual emissions of more than 750 million cars.⁷ In 2018 alone, 200 billion kilowatt-hours of electricity have been saved in buildings, which means 150 million metric tons of greenhouse gas reductions. However, such an amount of saving, about 7 % of the total electricity demand in buildings, is not large enough to stop global warming.⁸

Load reduction can make a tangible impact on decreasing energy demand. Without paying much attention, the average U.S. residential buildings use about 25 % more electricity than necessary, which in turn increase 18 % of total fossil fuel consumption.⁸ Many commercial buildings utilize building management system (BMS) to automatically control and monitor the HVAC and lighting system, however, low return on investment of such system have prevented consumers from adopting this technology in the residential sector. For example, 20 % energy reduction would save \$15 per month for the average U.S. household,⁹ which may not justify the difficulties of optimizing HVAC operation on a daily basis or installation of a new control system. At the national scale, the amount of savings translates to over 100 billion kWh or \$15 billion annually, which is equivalent to electricity consumption in California for half a year.¹⁰ Moreover, 1.12 billion tons of carbon dioxide can be prevented from being released into the air each year,^{11,12} which is about twice of the California annual carbon dioxide emissions.¹³

Heating, ventilation, and air conditioning (HVAC) equipment is an essential part in typical house and keeps buildings in a comfortable temperature range. Figure 3 shows a pie chart of residential energy usage in the U.S. The largest source of residential energy consumption in the U.S. is HVAC, which accounts for about 50% of a typical house's total energy usage¹⁴ leading to an average cost of about \$650 every year.

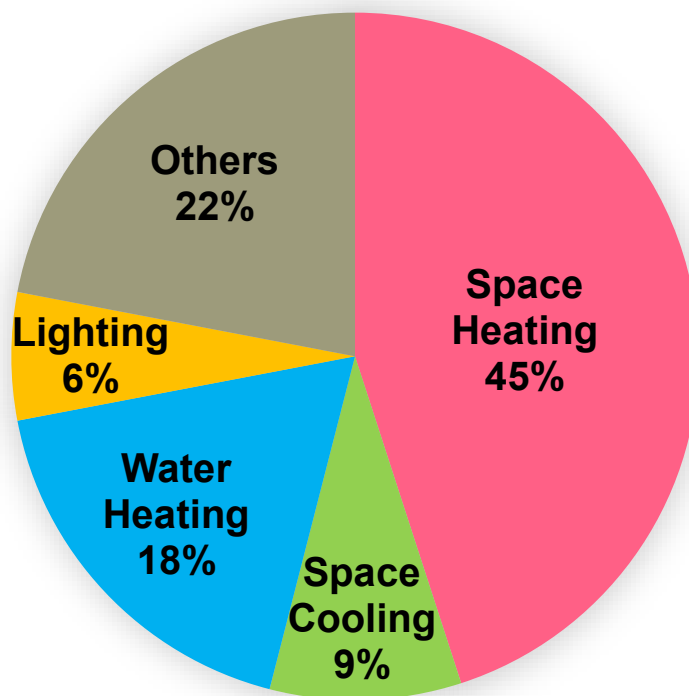


Figure 3: Annual household site end-use consumption in the U.S. (data from Energy.gov¹⁴)

Usually, the measurable amount of energy in HVAC is wasted due to unnecessary operation, such as overheating/overcooling or operation without occupants. Based on the recent paper by Nguyen and Aiello,¹⁵ energy conscious behaviors in residential homes can lead to 33 % less energy consumption compared to the design point, and 50 % less energy consumption compared to those demonstrating wasteful behaviors

as shown in Figure 4. Optimizing the operation schedule of HVAC to match the occupant presence schedule can reduce energy consumption significantly. Occupancy information will play a critical role for automatic control of lighting and HVAC systems. A report from the World Business Council for Sustainable Development as well as researchers suggest that minimizing HVAC operation in the absence of occupants can reduce energy consumption by 10-20 %.^{9,16} To engage more people into energy saving practices, an economic solution must be created that monitors the occupancy pattern, controls the HVAC system, and thereby saves energy without requiring constant attention by household residents.

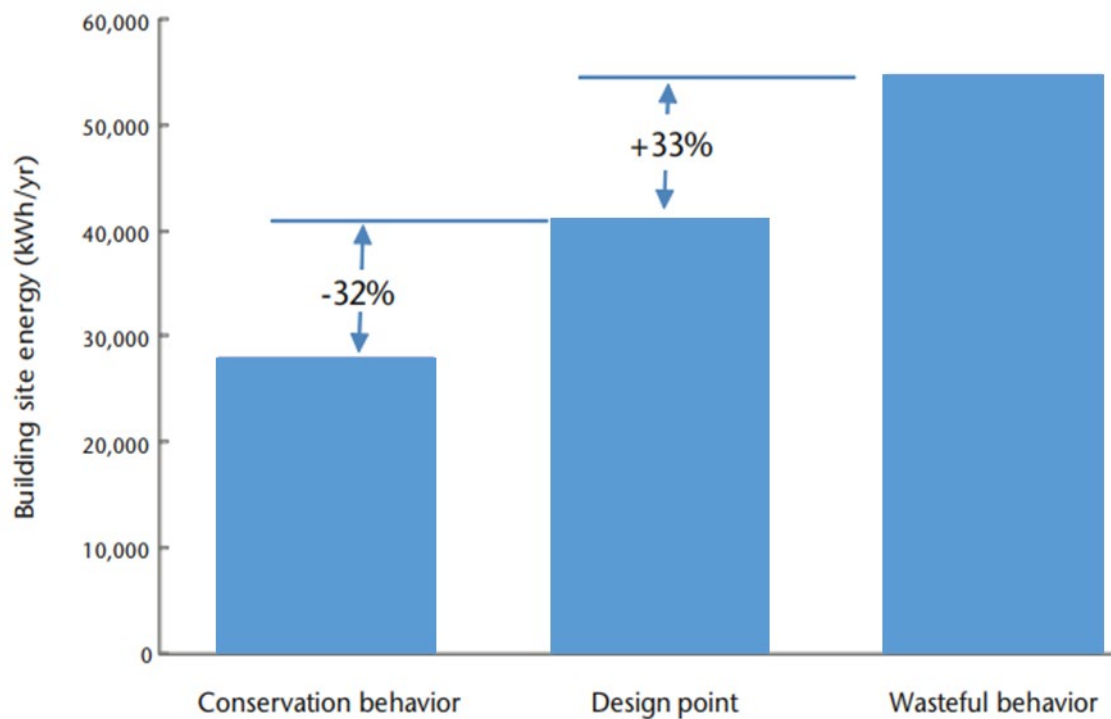


Figure 4: Energy-unaware behavior uses twice as much energy as the minimum that can be achieved (data original from WBCSD¹⁶)

Recent advancements in Cyber-Physical System (CPS) for smart home devices integrates home devices, sensors, and smart phones into one central system, which is able to control these devices by a home automation control algorithm based on sensor information. Such a system can potentially engage more people in energy conscious behavior without affecting their comfort level or lifestyle. Due to the lack of quality occupancy information, most of the existing smart home products on the market are only able to remote control the HVAC system, which also require energy conscious behaviors of users. This thesis provides a general solution of occupancy detection in residential buildings. Moreover, a co-simulation platform is explored to assess the energy saving impact and economic benefits of occupancy information in a residential house.

1.1. Review of Cyber-Physical System

A Cyber-Physical System (CPS) is a system in which a physical actuator is controlled or monitored by a sensor-based algorithm. CPS is a combination of networking, computation, and physical processes. The main parts of the CPS are sensors, actuators, communication technologies, and embedded computing units. Sensors are used to monitor the operating status of a system by collecting data like temperature, flow rate, or energy consumption. Sensors capture all necessary information that is required by the computing unit to control the system. Computing units process incoming data with some embedded algorithms and send out the control signal to actuators. Actuators transform the digital control signals into physical actions. Figure 5 shows a basic structure of a CPS.

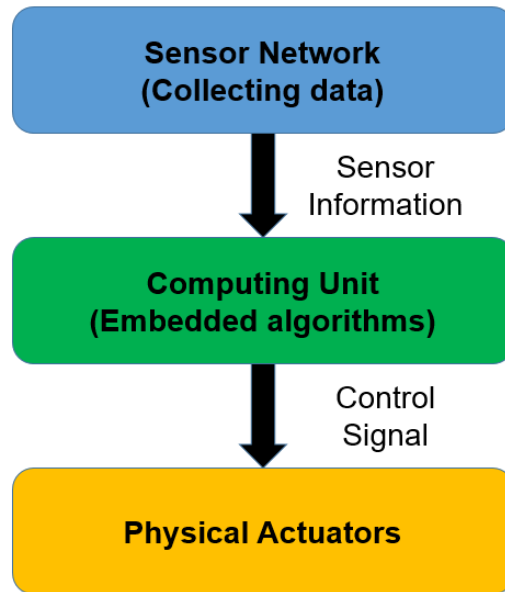


Figure 5: Overall structure of a Cyber-Physical Systems

Recently, the application of CPS has been expanded to multiple areas. Liang et al.¹⁷ developed a CPS with big data technology to optimize the operation schedule of machines in a workshop. According to the results, about 40 % of energy-savings and 30 % of productivity improvement have been achieved by implementing the new CPS. Thomas et al.¹⁸ designed a cyber-physical production system that can handle the demand of real-time evaluation and analysis of highly complex production systems in the modern manufacturing industry. The system showed the potential of reducing the time of current approaches used in the manufacturing process by 74 %. Fatimah et al.¹⁹ proposed a smart waste collection system to control and to communicate all waste management components for semi-urban cities in developing countries. The system significantly saves time and cost of the entire collection process by automatically sending waste to the conveyors based on the waste characteristics.

Over the last few years, CPS started to relate with energy management system. Gunduz and Jayaweera²⁰ proposed a cyber-physical integrated power system, including multiple photovoltaic components, which can efficiently reduce the fluctuation of photovoltaic power plants for power grids. Behl et al.²¹ designed an open-source CPS for energy management, called DR-Advisor, which supports data-driven modeling and control with rule-based algorithms. Grigore et al.²² developed a CPS for thermal optimization via power load monitoring, predicting and smart grid integration.

As the premise of building management systems is to save as much energy as possible while ensuring occupants' comfort, Balafi et al.²³ stated that occupancy information played a fundamental role in smart building management. Liang et al.²⁴ also suggested occupancy data should be used to improve the accuracy of building energy prediction since occupancy information is highly correlated with energy consumption. How to sense and measure occupancy information in a building becomes the basis of such cyber-physical based building management system.

1.2. Review of Occupancy Sensors

Occupancy is not just a simple measurement of presence of users. Labeodan et al.²⁵ has identified six spatial-temporal properties of occupancy measurement for effective demand-driven energy system control: presence, location, count, activity, identity, and track. While presence, count, and location are important to decide whether to run HVAC system or not, identity, activity, and track can be exploited to better determine

HVAC operational parameters to satisfy personalized comfort level and reduce energy consumption. Commercially available solutions for occupancy measurements include passive infrared detection systems (PIR), electromagnetic sensors, CO₂ sensors, and image detection systems.

Passive infrared detection systems (PIR): PIR sensors detect a moving, warmer-than-background object passing through a fixed view angle.²⁶ This type of sensor is limited to occupant driven control of lighting systems due to several reasons: 1) the sensors output is binary (can only provide the presence and location information). 2) The sensors require a technician to fine-tune the set-up and wiring design based on the house floorplan. 3) The sensors have been known to also be falsely triggered by HVAC system heat currents, and as a result, the accuracy of PIR sensors is low.

Electromagnetic signal (EM) detection systems: EM detection systems use wireless fidelity (Wi-Fi), global positioning system (GPS), radio frequency identification tags (RFID) or Bluetooth tracking technologies. It requires occupants to carry a device that is powered up all the time, increasing user inconvenience.²⁷ Because building users often have more than one EM signal broadcasting device, this will cause false registrations of presence, location and count.

CO₂ based detection systems: CO₂ sensors measure the concentration of gases in a space in parts-per-million (PPM). Multiple research papers utilize CO₂ sensors for the measurement of occupancy in buildings for demand-driven control of HVAC systems.^{28–31} The amount of CO₂ in a space can be employed in determining

occupancy information on presence, location, count, and user activity. Drawbacks of CO₂ sensors include high-cost, disruptive installation, delayed detection, and instability due to unpredictable airflow patterns.

Image detection systems: Image recording devices such as video cameras are often used in buildings for security purposes. The application of image detection system is rarely used in residential building for a number of reasons: 1) It requires a direct line of sight; 2) Advanced signal processing and expensive explicit hardware is required by the system to analyze images and output occupancy information, thus raising costs; 3) User privacy is a major issue of concern and worry.

Stancil et al.³² and Trivedi et al.³³ utilize a multi-camera network to detect and measure the number of the occupants in the building. Erickson et al.³⁴ also developed a multi-camera based system for occupancy prediction and real time occupancy monitoring and tracking. A variety of multi-camera based occupancy measurement systems are available in commercial buildings. However, application of such systems are limited in residential buildings. Erickson et al.³⁵ also reported that the two main limitations are the privacy concerns and the high cost of such system.

A brief summary of all the aforementioned individual occupancy sensors is listed in the Table 1. The price row in the table only shows the price of transducer I found in Digi-Key.³⁶ The actual sensor package may require more cost on data processing and communicating. Almost all of the individual detection systems are unable to provide quality occupancy information due to multiple reasons. With the development

of data mining, research efforts have begun to use machine learning models to predict occupancy information from non-intrusive environment sensors.

Table 1: Summary of existing occupancy sensors in the market.

Typers of sensors	PIR	EM	CO ₂	Image Detection
Privacy	✓	✓	✓	✗
Installation	✗	✓	✗	✗
Convenience	✓	✗	✓	✓
Price	\$2	\$8	\$13	\$9

1.3. Review of Occupancy Detection via Machine Learning

Over the past decades, multiple machine learning (ML) algorithms and probabilistic models have been developed to relate occupancy information with environment sensors such as temperature, CO₂ concentration, light intensity, and noise level. Jiang et al.³⁷ applied an Extreme Learning Machine to estimate and predict the number of occupants using CO₂ concentrations and verified the model in an office room. The model can estimate the number of occupants with a three-person margin of error at 89 % accuracy. Page³⁸ used two years of data to train Markov chain models for occupancy information prediction. This model produced a time series of the occupancy state that considered the randomness of human behavior. Candanedo and Feldheim³⁹ implemented a Decision Tree along with three other classification

models, including Random Forest, Gradient Boosting Machines, and Linear Discriminant Analysis to predict the occupancy status in a room. The result showed that high accuracy (around 93 %) was found when using temperature and light data with the Linear Discriminant Analysis models. Yang et al.⁴⁰ compared the performance of six ML techniques (Support Vector Machine, K-Nearest Neighbors, Artificial Neural Network, Naïve Bayesian, tree augmented Naïve Bayes network, and Decision Tree), and concluded the Decision Tree technique yielded the best overall accuracy.

Previous researches focused on implementing ML methods using the data collected from sensors in one or more specific rooms or buildings. Nevertheless, due to the large variety of buildings (e.g., floorplan, material, location, and orientation), the algorithms and the combination of sensors suggested by each work are different and may only be valid for specific buildings or room configurations. The fact that different algorithms were used for different studies indicates that the previous approaches were not transferrable. Table 2 shows the diversity of machine learning algorithms and sensor combinations suggested by previous approaches.

Table 2: Summary of machine learning algorithms and type of sensors for occupancy measurement.

Author	Sensor	Machine Learning Algorithms
Abade et al. ⁴¹	Temperature, CO ₂ , Light intensity, Sound	Logistic Regression(LR), Support Vector Machine(SVM), Artificial Neural Network(ANN)
Vafeiadis et al. ⁴²	Power consumption, Water consumption	SVM, Decision Trees(DT), Random Forest (RF), Back-Propagation Network (BPN)
Ortega et al. ⁴³	PIR, Pressure (piezo), Contact	SVM , multiclass SVM
Yang et al. ⁴⁰	Light, Sound, Motion, CO ₂ , Temperature, Humidity, PIR	SVM, KNN, ANN, naïve Bayesian(NB), tree augmented naïve Bayes network (TAN), DT
Tutuncu et al. ⁴⁴	Light, Temperature, Humidity, CO ₂	7 different ANN (Quick Propagation, Conjugate Gradient Descent, Quasi-Newton, Limited Memory Quasi-Newton, Levenberg-Marquardt, Online Back propagation, Batch Back propagation)
Candanedo and Feldheim ³⁹	Light, Temperature, Humidity, CO ₂	Random Forest (RF), Gradient Boosting machines (GBM), Linear Discriminant Analysis (LDA), Classification and Regression Trees (CART)
Wang et al. ⁴⁵	Wifi probe: unique hardware address (UUID)	Markov Chain model
Hailemariam et al. ⁴⁶	Light, Power consumption, CO ₂ , Sound, Motion	Decision Trees
Wang et al. ⁴⁷	Temp, Humidity, CO ₂ and Wi-Fi probe: MAC address	K-nearest neighbors, SVM, ANN

Many ML approaches have been explored to extract occupancy information from the sensor data and handle the randomness of human behaviors. However, the accuracy of ML models is highly dependent on the installation condition of the sensors, which

is impossible to keep consistent across different residential buildings. For instance, the temperature data of the hallway can highly relate to occupancy information in the testing building but has a low correlation in another building (different floorplan, material, location, or orientation). Collecting and inputting temperature data of the hallway would not help ML models make the correct prediction since this information could not represent the occupancy information anymore.

For an arbitrary house, conventional ML models often require more than enough sensors to ensure that the collected data can represent the occupancy patterns. This poses great challenges to wide adoption of such technologies. Another challenge of ML is that it needs adequate data to extract appropriate features and train the model, which is time consuming and may increase the cost of the system.

1.4. Two-layer Occupancy Detection Scheme

A recent report from the U.S. Department of Energy (DOE)⁴⁸ states that incorporating domain knowledge had been proven as an effective approach to help supervised and unsupervised ML models. Wagstaff et al.⁴⁹ and Webber⁵⁰ suggested that domain knowledge can improve accuracy and simplify the choice of representative features. This thesis considers the role of resident behavior during occupancy and during the changes to occupancy, regardless of the floor plan, material, location, and sensor orientation. This knowledge was used to develop a novel sensor system that uses domain knowledge to detect specific human behaviors that are highly correlated with human occupancy. For example, a user needs to touch the main door handle to enter

or leave the house. Water is more likely to be drawn from the faucet when a house is occupied. Since human activities, like door open or water usage, are independent of building floor plan or installation conditions, such an approach can be readily transferrable to any residential building with a limited number of sensors.

However, human activities, such as water usage, are not always related to occupancy information, which may mislead occupancy information in some cases. For example, scheduled washing machine operation or door handle touching by delivery persons may lead to false-positive occupancy information. In order to avoid such misleading cases, machine learning algorithms are used to retrieve the actual occupancy information from all human activities that are detected in a short period. Figure 6 illustrates the overall schematic of the occupancy detection model. Multiple economical sensors collect the non-intrusive environmental information (temperature and motion) from different locations in the house. The sensor data is consumed by white-box detection models for recognizing a set of human activities (door handle touch, water usage, and motion near the door area), which are highly correlated with the change of human occupancy. Then a series of human activities happens in a short period are summarized into an event, and a ML based classification model is incorporated to identify whether it's an entering event, leaving event or no change event. After combining with the previous occupancy state, the actual occupancy information can be retrieved accurately. The whole two-layer occupancy detection scheme is cost effective, non-invasive, non-intrusive, and user friendly, which is much easier to be adopted by more residential users.

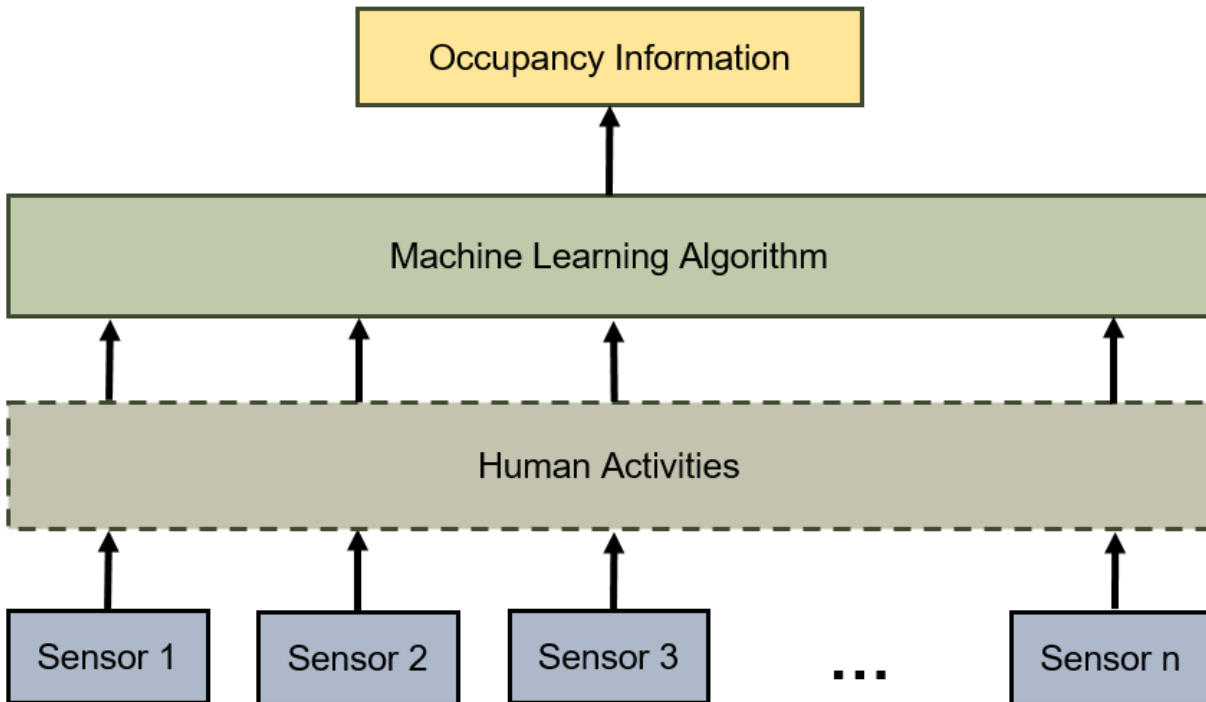


Figure 6: Representation the schematic of data processing from the environmental sensors to the final predicted occupancy information.

1.5. Energy Saving Impact of Occupant Driven Thermostat

Occupancy information can lead to a huge potential for energy savings in residential buildings. Most of the existing smart thermostats advertised that the average energy saving is between 10 % and 30 % by comparing the energy usage of two groups of houses: installation group and a control group of non-participant homes.⁵¹ The thermostat savings in any given home can vary significantly from the average values, and the experimental validation takes a long time and resources. Figure 7 shows the distribution of energy saving impact of existing smart thermostats. Depending on the

location, the true energy savings attributable to the thermostat has a lot of variability, including that the gas usage in many homes even increased.

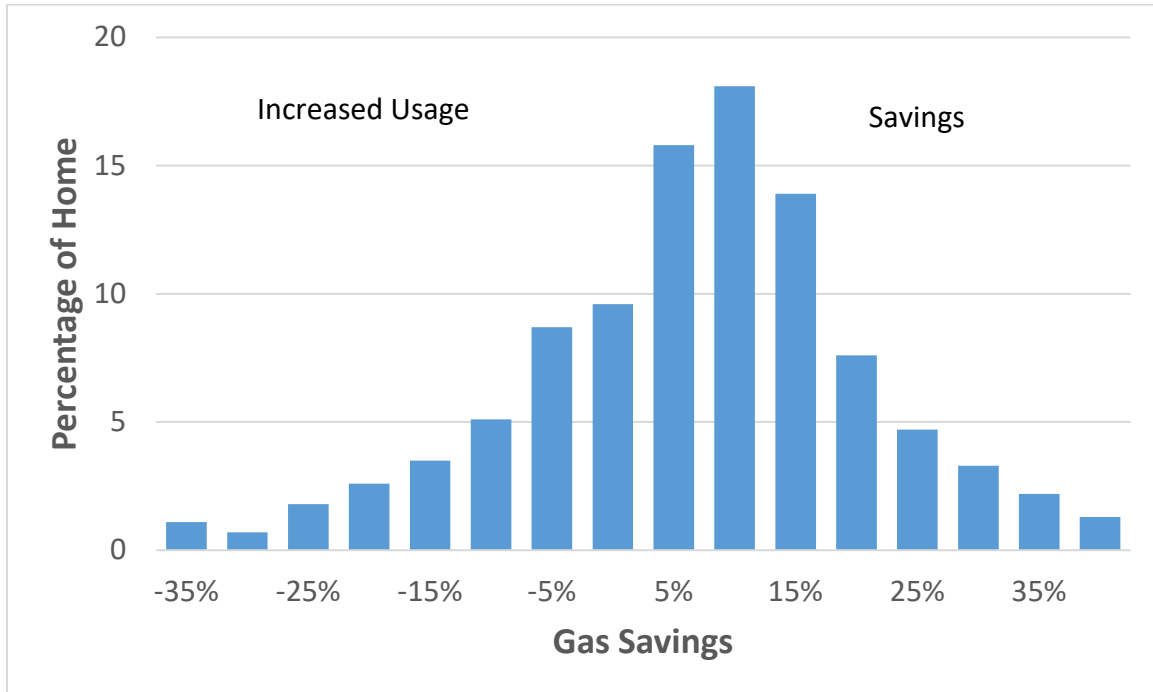


Figure 7: Distribution of gas savings for existing smart thermostats. (data original from NEST white paper⁵¹)

This work evaluates the energy saving impact of an occupancy driven thermostat in a residential building via simulation. Some existing building energy software,⁵² such as EnergyPlus, calculates the thermal mass and resistance of a given building model based on its floorplan, material and wall structure. After inputting local weather, HVAC system configuration, and setpoint control strategy, EnergyPlus is able to estimate the energy load in the building. However, such software cannot readily consider a newly developed occupancy-based thermostat control algorithm or the random nature of occupancy. To overcome this limitation, an occupancy simulator and HVAC controllers were developed. The occupancy simulator provides the occupancy

information based on a stochastic model, which can show the variation of occupancy pattern. The HVAC controller changes the heating/cooling setpoint based on the occupancy information as well as the outdoor temperature. Since the interoperability of the building energy software is restricted, a co-simulation platform was utilized to integrate self-developed simulation entities with existing building energy software.

This paper is not the first attempt to use co-simulation to overcome the limitation of the single building energy software. Hensen⁵³ coupled a general-purpose building simulation package with a CFD approach to depict the heat and air flow in buildings. Zhai and Chen⁵⁴ developed their integrated building design tool, E+MIT_CFD, which incorporated a CFD program (MIT-CFD) into EnergyPlus. Trcka⁵⁵ integrated a building model in EnergyPlus and an air system model in TRNSYS through both loose coupling strategy and strong coupling strategy. Co-simulation with more than two simulation entities is not well explored to be readily used by others. Few works^{56–58} exploited Building Controls Virtual Test Bed (BCTVB) as a middleware to couple more than two simulation entities. However, BCTVB can only work with limited types of simulation tools and is not compatible with other languages such as Java or Python, which are often used in recent research works or products. Hence, the prior works are not readily scalable and cannot consider the random nature of occupancy information as well as new thermostat models. In order to address the aforementioned issues, researches always prefer to use the simulation entities that are best suited to model various aspect of control systems. This work exploits an open source co-simulation platform that utilizes the IEEE standard (High Level Architecture (HLA)) to integrate multiple heterogeneous simulation entities. Six HVAC system control

strategies were explored based on three types of thermostats (always on, schedule based, and occupancy driven) as well as two setpoint control algorithms (fixed setpoint and adaptive control). EnergyPlus was integrated into the co-simulation platform, which evaluated energy consumption and indoor temperature of a residential house in five climate conditions. A comfort level evaluator was also devised and utilized to estimate the user's feedback in the building.

1.6. Thermal Comfort Standard and Adaptive Control

Users' comfort level should not be sacrificed for the sake of energy saving, which will affect occupants' health.⁵⁹ Tanabe⁶⁰ observed a near-infrared spectroscopy based brain imaging, and he found a reduction in productivity as comfort level decrease. Seppanen⁶¹ developed a quantitative relationship between work performance and the cost of HVAC electricity. The ratio of productivity gains to energy used by HVAC varied from 32 to 120. The human comfort level is not only based on the indoor temperature, it is also highly influenced by many other factors, including but not limited to humidity, cloth level, metabolism, and outdoor temperature. Thermal comfort standards have significant impacts on the energy consumption of HVAC systems by affecting the cooling and heating setpoints. The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) Standard 55 introduced two thermal comfort criteria: (1) predicted mean vote (PMV) comfort zone limits^{62,63}; (2) adaptive comfort.^{62,64} The PMV comfort zone model is the most often used model in the last 20 years, which is based on human body energy balance and an empirical fit to

thermal sensation. Recently, many researchers^{65–67} conducted multiple surveys and questioned the applicability of the PMV to general public. Howell⁶⁵ showed that numeric scale of thermal sensation for PMV model was not entirely valid, so the prediction is warmer than how users feel. Croome et al.⁶⁶ stated that the steady-state assumption in the PMV model made the predicted conditions less comfortable. Humphreys⁶⁷ analyzed the original database used to derive PMV model, and found that the PMV model is only reliable near the comfort condition. The error was getting more significant when the thermal condition moved further from the neutral. Since the PMV model is based on the thermal sensation of college-aged students in air-conditioned buildings in moderate climate zone, it cannot accurately evaluate human thermal comfort level in any other kind of building in different climate zones. As a result, a field study⁶⁸ showed that the energy consumption of PMV model is unnecessarily high without providing a better comfort level.

In contrast, the adaptive comfort model defines the indoor human comfort level in residential house based on mean outdoor temperature, and therefore the issues associated with the PMV model can be avoided. For example, a room temperature of 20°C makes people feel chilly in the summer but warm in the winter. Based on the adaptive comfort model, thermal comfort in a building can sometimes be achieved only with natural ventilation during warm season, accompanied by significant energy saving. Although adaptive comfort was originally designed for naturally ventilated buildings, it is also recommended for use with buildings that have HVAC systems.^{62,64} Parkinson⁶⁹ also validates the ASHRAE 55 adaptive comfort standard with a larger and more representative dataset (over 100,000 thermal comfort field data from all

around the world). The results showed that adaptive comfort processes are relevant to the occupants of all buildings, both air conditioned and naturally ventilated.

Multiple studies suggest using an adaptive or predictive HVAC control system, which is summarized in a review paper by Song et al.⁷⁰ Among those, Lute and Paassen⁷¹ identified the adaptive fuzzy controllers as one of the most capable models for buildings. The fuzzy control does not require the exact information of building system, however some system parameters such as fuzzification, inference, and defuzzification, need to be tuned. Nesler⁷² suggested a parameter estimation method based on Recursive Least-Squares estimation (RLS). Tigrek⁷³ introduced a model using gradient descent for the estimation of the parameters. By utilizing the estimation methods with HVAC control system, the model can calculate the system parameters automatically. Such methods require a data collection period and the long run time of the system. These previous works target commercial buildings in which the controllability of the HVAC is far more complicated than residential building. ASHRAE Standard 62.2 is the only recorded document which considers adaptive HVAC control system in a residential house.⁷⁴ The standard focuses on the ventilation system to provide acceptable indoor air quality in residential buildings. To the best of my knowledge, there is no documented study providing the energy-saving impact of an adaptive HVAC control system in residential buildings.

1.7. Thesis Objectives

The aim of this work is to estimate occupancy presence information (occupied/vacant) in a residential building and use it to save energy. The first objective is to develop a cost-effective and non-intrusive occupancy detection scheme for residential buildings. Another objective is to design a simulation tool to evaluate energy saving impact and economic benefits of an occupancy-driven thermostat.

For the first part of this thesis (Chapter 2 & 3), I developed a preliminary approach of occupancy detection utilizing a two-layer scheme based on data obtained from non-intrusive sensors (temperature and motion). (1) Human activity layer (lower layer): transient and steady-state heat balance equations have been derived to develop heuristic models, which convert continuous temperature data into whether a specific human activity happens. Since the temperature data is highly dependent on season, weather and other factors, the data is first normalized with ambient temperature. Then the normalized temperature change is compared with a threshold, which is calculated based on the previous detected activities. When the normalized temperature change exceeds the threshold value for a certain period, the human activity is considered to be detected. (2) Machine learning layer (upper layer): ML models (Random Forest, Decision Tree, K-Nearest Neighbor, and Support Vector Machine) are introduced to determine the real-time occupancy information from a series of human activities. The recognized human activities occur over a variable period of time, and the resulting data cannot be readily applicable to existing machine learning models due to varying input matrix sizes. Hence, a fixed length format was devised to describe the event

regardless of its total duration. From the result, a valid estimation of occupancy information can be predicted by using the Random Forest algorithm with high accuracy and F1-score (both >98 %). Moreover, the number of sensors can be further decreased to three without risking the validity of occupancy prediction. The key contributions are: (A) providing a new method for occupancy detection in residential buildings with a limited number of low-cost and non-intrusive sensors, and (B) proposing a two-layer occupancy detection scheme with improved generality by incorporating domain knowledge into ML algorithms.

The second of part of this thesis (Chapter 4) explored a co-simulation platform to assess energy saving impact and economic benefits of occupancy information in a residential house. Occupancy-driven thermostat control algorithms are developed. An occupancy simulator is devised and utilized to consider the random nature of the occupancy in a typical single family residential house. Six HVAC system control strategies are explored based on three types of thermostats (always on; schedule based; and occupancy driven) as well as two setpoint control algorithms (fixed setpoint; and adaptive control). A building energy software, EnergyPlus, is integrated into the co-simulation platform, which evaluates energy consumption of a residential house in five cities (Fairbanks, New York City, San Francisco, Miami, and Phoenix) with distinctive climate zones. User's comfort level is evaluated using the adaptive model for the six different control strategies in each location. The result showed that the occupancy-driven control algorithm can save about between 11% and 34% of energy without significantly risking the occupant's comfort level. The key contributions are: (A) provide co-simulation tool and analysis on energy saving impact with occupancy-

driven HVAC control algorithms, (B) consideration of random nature of occupancy information and users' comfort in a residential house, and (C) devise an adaptive control approach to further save energy in a residential house.

The rest of the paper is organized as follows: Chapter 2 explains the economic non-intrusive human activity detection models by introducing the door handle touch detection model, the water usage detection model, detailed experiment set-up, and the validation of every single detection models. Chapter 3 presents the ML based data fusion scheme by introducing the ML algorithms used in this thesis, the detailed experiment setup, the data processing, and its validation. Chapter 4 presents and discusses the energy saving impact of occupancy information and adaptive control by introducing the HLA based co-simulation platform, the simulation entities, and economic analysis of occupancy-driven thermostat. Finally, Chapter 5 covers conclusions and future works.

Chapter 2. Economic Non-Intrusive Human Activity Detection Models

The recognition of human activities is a high-interest task for medical applications. For example, doctors usually require patients with heart disease to follow a well-defined exercise routine as part of their treatments. Therefore, recognizing activities such as walking, running, or sleeping can provide more straightforward feedback to the caregiver about the patient's behavior. A similar idea can be used for occupancy detection. Some specific human behaviors, such as open door, close door, water usage, are more general and highly correlated with occupancy information. In commercial sector, some existing occupancy sensing products, such as Traf-sys,⁷⁵ Sensource,⁷⁶ and Density,⁷⁷ utilizing overhead cameras and image detecting technologies to track and count individuals entering and existing events. However, video cameras are considering as the most privacy-intrusive sensors in residential buildings.⁷⁸ For example, V. Srinivasan et al.⁷⁹ present a new attack that can observe private activities in the home, such as showering, toileting, and sleeping, even if all the data transmissions are fully encrypted. And these attacks are not only hypothetical; a report⁸⁰ from CSO reveals over 73,000 video cameras were found to be streaming their surveillance footage on one website in 2014. Like Density,⁷⁷ some products already added some privacy protection features to ensure not capturing personally identifiable information. Such approaches will also increase the cost of the whole occupancy detection system. If we adopt this product in a residential house, the

payback period of a one-sensor system is more than six years. Hence the proposed approach deployed temperature sensors (K-type thermocouple) and developed multiple detection models to recognize human activities from temperature data, which makes the whole sensor non-intrusive, non-invasive and cheap to collect.

2.1. Door Handle Touch Detection

The door handle touch detection model is based on a simple idea. If someone enters or leaves a room, he or she need to touch and pull the door handle. The temperature of the door handle will change due to the heat transfer with human hand. By measuring and recording the rate of temperature change, the event of a person touching a door knob/handle can be detected.

Assuming spatially uniform temperature distribution, the energy conservation equation for a door handle is shown in Eq. (2.1),

$$mC_p \frac{dT}{dt} = \dot{Q} \quad (2.1)$$

where T is the door handle surface temperature, m is the mass of the door handle, C_p is the specific heat capacity of the door handle, \dot{Q} is the amount of heat transferred to the door handle, and t is time. m and C_p are assumed constant in a very short time frame, around 1 second.

With no touch, the door handle will only gain/lose heat to the environment through natural convection. The temperature of the door handle will rise or drop slowly. As a result, the rate of temperature change will be relevant small.

For a touch event, \dot{Q} contains not only natural convection from the environment but also heat conduction from a hand. Since skin temperature of a healthy human is around 34 °C,⁸¹ which is often higher than ambient temperature, the heat transfer via direct contact is much greater than the convection and thereby the rate of temperature change is significant during the touch event.

2.1.1. Detection of Temperature Change

As the accuracy limits of the thermocouple, the collected temperature profile has a huge fluctuation, about 0.5 °C, even without touch event (Figure 8). This work proposed to use the moving average method which helps smooth out the temperature data curve by filtering out the noise from random temperature fluctuations. The formula is shown below Eq. (2.2),

$$T(t) = \frac{1}{2M + 1} \sum_{t-M}^{t+M} T_{raw}(i) \quad (2.2)$$

where T is the temperature after moving average method, T_{raw} is the raw temperature data collected from thermocouple, M is the moving average coefficient and t is time. The temperature data after moving average is determined by M data points before, M

data points after and the data point at that moment. With the moving average method, the temperature data profile demonstrates the temperature trend without huge fluctuation (Figure 8).

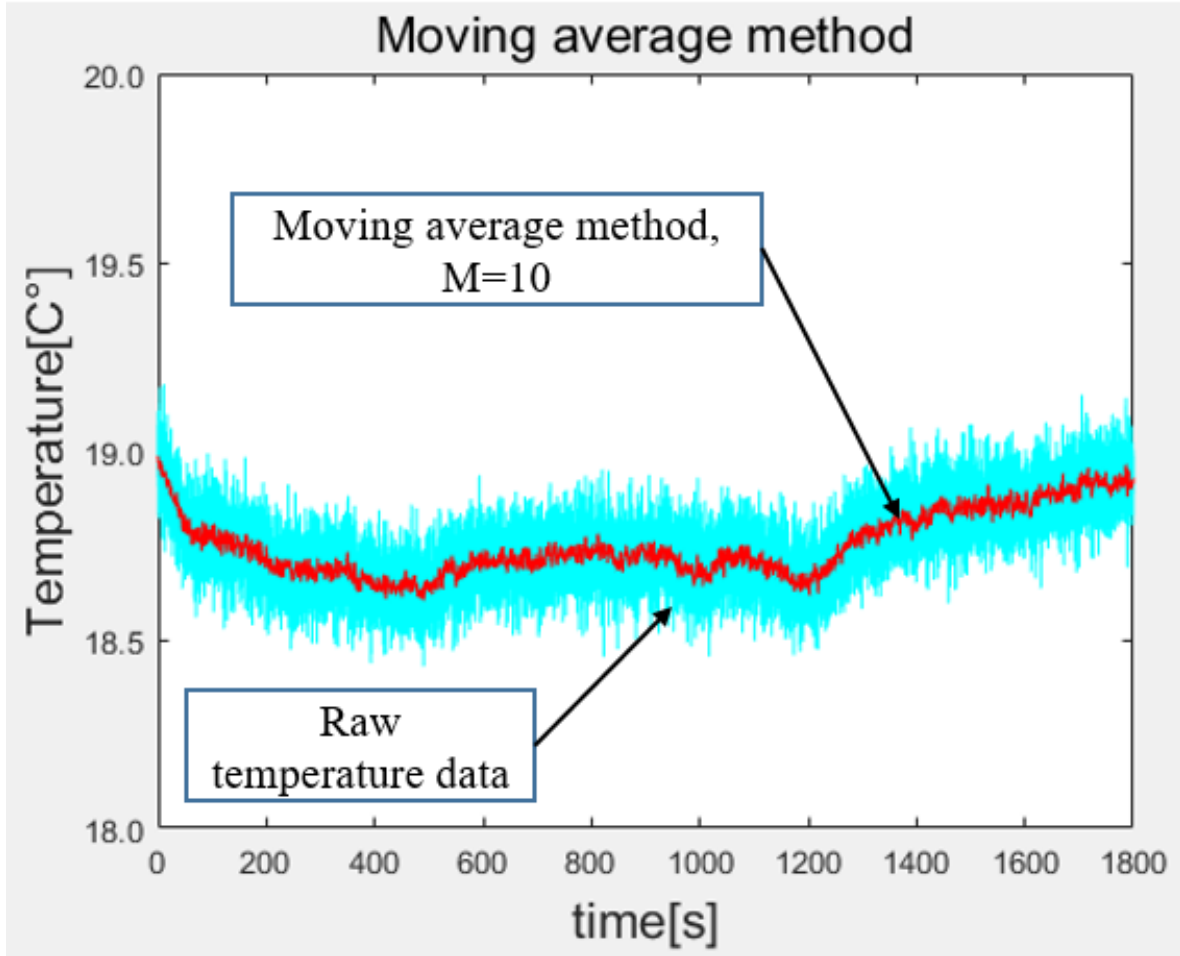


Figure 8: Raw temperature profile measured by K-type thermocouple with huge fluctuations, about 0.5 °C, compare with temperature profile after using moving average method which keeps temperature trend, filtering out fluctuation.

2.1.2. Criteria Determination

The detection method assumes differentiated human touch from natural temperature increase due to the ambient temperature increase. Temperature change rate will be

different for these two events. For the no touch event, temperature of door handle changes slowly. But for the touch event, the temperature change rate is relatively high. As discussed in the previous section, the temperature change rate is dependent on \dot{Q} , the total amount of heat transfer to the door handle. \dot{Q} contains natural convection from the environment and heat conduction from the human fingers.

During a touch event, the convection term is negligible as the initial temperature of the door handle T_i is assumed to be equivalent to the ambient temperature T_∞ . The temperature change rate is only affected by the heat conduction from the finger. The heat balance equation is shown in Eq. (2.3).

$$mC_p \frac{dT}{dt} = hA(T_{finger} - T(t)) \quad (2.3)$$

where T_{finger} is the human finger temperature, A is the touching area, and h is the overall heat transfer coefficient. By integrating Eq. (2.3), an expression of normalized temperature change rate is illustrated Eq. (2.4),

$$\frac{dT}{T_i - T_{finger}} = e^{-b\Delta t} \quad (2.4)$$

$$b = \frac{hA}{mC_p} \quad (2.5)$$

where m is the mass of the door handle.

Because of the diversity of the ambient temperature in different regions of the world, it is difficult to preset a single ambient temperature. And temperature change rate is dependent on $T_i - T_{finger} \approx T_\infty - T_{finger}$ which directly influenced by ambient temperature. Instead of temperature change rate, this work used normalized temperature change rate, $\Delta\dot{T}$, to detect touch events. The conversion equation shows in Eq. (2.6).

$$\Delta\dot{T}(t) = (dT(t)/dt)/(T_{finger} - T(t - \Delta t)) \quad (2.6)$$

And several input variables such as finger temperature, material, volume and touching area of the door handle are case-sensitive. So it is nearly impossible to preset a fixed temperature change rate threshold for all different local events. So the touch detection model utilizes previous touch data to setup the threshold level. The temperature change rate of the door handle is collected and recorded. After a new touch event has been detected, the threshold will be updated based on the mean and standard deviation. of $\overline{\Delta\dot{T}}$, average normalized temperature change rate, of previous touch events Eq. (2.7),

$$Threshold = \frac{\overline{\Delta\dot{T}}}{2} - z \frac{\sigma_{\overline{\Delta\dot{T}}}}{\sqrt{N}} \quad (2.7)$$

where $\overline{\Delta T}$ is the average normalized temperature change rate of all previous touch events, $\overline{\Delta T}$ and $\sigma_{\overline{\Delta T}}$ is the mean and standard deviation of it, N is the number of previous touch events, and z is the confidence level coefficient of this method.

False negatives should be avoided for the residential application – when a sensor cannot detect the occupancy, this leads to insufficient air conditioning/ventilation and thereby creates health concerns. This also harms the user adoption of the technology. “False positives”, on the other hand, reduces the energy saving by keeping the HVAC system on when the space is not occupied, and are less detrimental to user adoption. So the “false negatives” need to be prevented to the best of the capabilities without highly increasing the risk of “false positives”. Assuming normal distribution, 95 % of values are above the threshold, which equals 1.65 standard deviations of the mean below the half of average maximum value.

2.1.3. Uncertainty Control with Data Collection

Because human touching is an action usually lasting more than one second, a touch event can be determined if the average temperature change rate in one second is greater than the threshold. Since the temperature data is collecting in real-time, the touch event can be detected in 1 second after the actual touch. And every time a new touch event has been detected, threshold value will be updated based on a new mean and standard deviation. With more and more touch data collected, the threshold will

stabilize at a certain level which can then allow us to reach more than 95 % accuracy at this local position.

Such a model requires to store a large amount of data since several temperature data points are appended to the dataset every second. However, only mean and standard deviation of the average normalized temperature change rates for previous touch events are needed to updating the threshold value. Instead of storing all previous temperature data points, only three values are recorded: the number of previous touch event, the mean, and standard deviation of the average normalized temperature change of previous touch events. The updated mean and standard deviation can be calculated from these values and new touch event data by using Eq. (2.8) and Eq. (2.9).

$$\overline{\overline{\Delta\dot{T}_{N+1}}} = \frac{1}{N+1} \left(N \cdot \overline{\overline{\Delta\dot{T}_N}} + \overline{\overline{\Delta\dot{T}_{N+1}}} \right) \quad (2.8)$$

$$\sigma_{N+1}^2 = \frac{(N-1)\sigma_N^2 + \left(\overline{\overline{\Delta\dot{T}_{N+1}}} - \overline{\overline{\Delta\dot{T}_{N+1}}} \right) \left(\overline{\overline{\Delta\dot{T}_{N+1}}} - \overline{\overline{\Delta\dot{T}_N}} \right)}{N} \quad (2.9)$$

where $\overline{\overline{\Delta\dot{T}_{N+1}}}$ is the average normalized temperature change rate of new touching event, $\overline{\overline{\Delta\dot{T}_{N+1}}}$ and σ_{N+1} are the updated mean and standard deviation value including the new touching event. As the standard deviation of mean reduces with increased number of samples, uncertainty will be reduced as more data is collected. The

flowchart of this method shows in Figure 9. The key parameters of this door handle touch detection model are listed in Table 3.

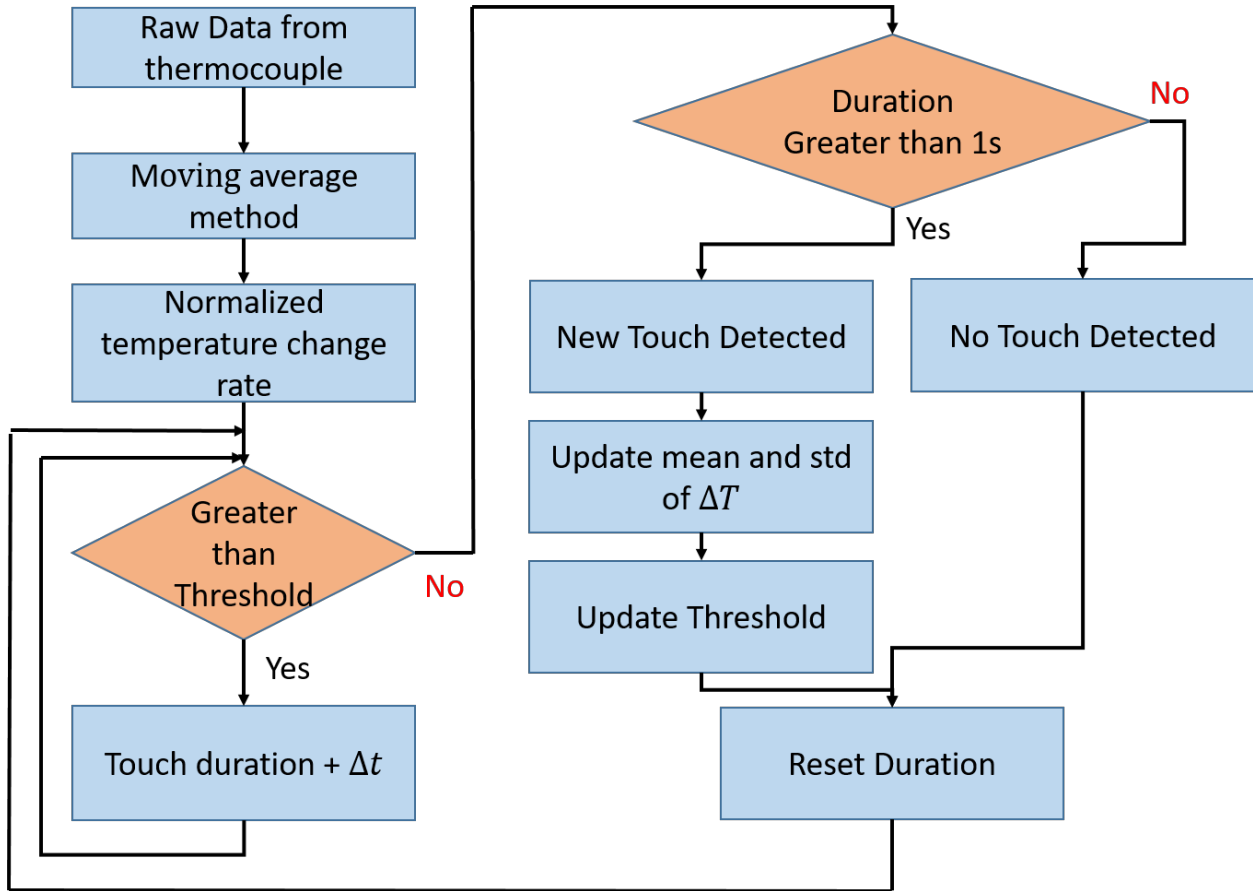


Figure 9: Flowchart of touch detection method.

Table 3: List of the key parameters used in door handle touch detection model.

Door Handle Touch Detection Model	
Moving average window	1 second
Hand temperature	34 °C
Minimal Activity Duration	1 second
Confidence level	95%

2.1.4. Experimental Set-up

A schematic view of the apparatus for measuring the temperature of the door handle is shown in *Figure 10a*. Two aluminum sheets (1cm×9cm) were screwed on both sides. The K-type thermocouple was clamped to the center of the aluminum sheet. Some silicon heat transfer compounds are applied on the thermocouples. Then, a piece of thermal insulation material is attached on the bottom side of the base aluminum sheet. And some two-sided tape is used to fix the experiment set-up on the door handle. The temperature data on the top aluminum sheet surface were measured and recorded with a NI Compact DAQ every 0.1 second. So the temperature change rate is equal to the actual temperature change divided by 0.1 second. The entire set-up is illustrated in *Figure 10b*. The uncertainty in temperature measurement is about ± 0.2 °C. All data by the temperature sensor was averaged by using the moving average method.

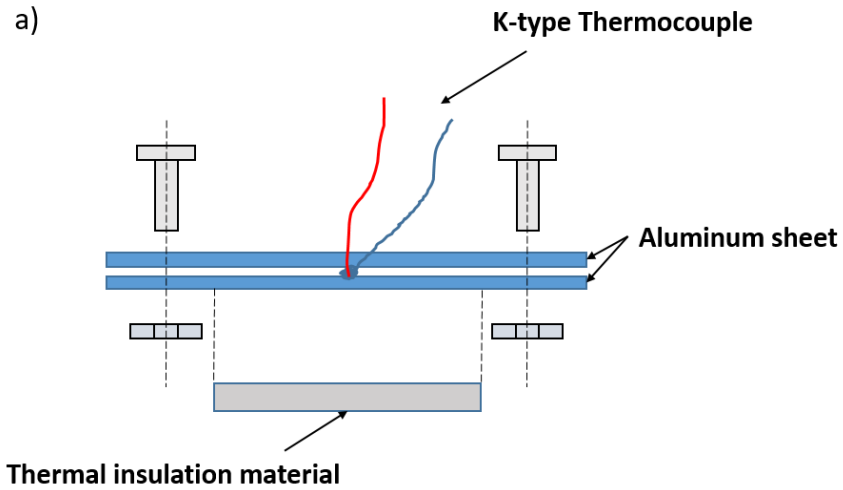


Figure 10: a) Schematic diagram of experimental set-up measuring temperature of this sensor with a K-type thermocouple. b) Experimental set-up attached on the top side of the door handle with some two-sided tape. Set-up is connected to a NI Compact DAQ to measure and record the temperature data every 0.1 second.

The mass of the aluminum sheet is 5 g, C_p is 0.9 J/gk, and touching area is the surface area of the top aluminum sheet (0.01 m \times 0.09 m). Assume ambient temperature is about 20 °C and finger temperature is 34 °C.⁸¹ It is difficult to estimate h precisely, an order of magnitude estimation is made by using Eq. (2.10).

$$h = O(k_{skin}/L_{finger}) \quad (2.10)$$

where k_{skin} is the thermal conductivity of human skin equals to 0.5 W/m·K,⁸² L_{finger} is half of the thickness of the finger, around 0.005 m.

By using Eq. (2.4) and Eq. (2.5) and information of this experiment set-up, the maximum temperature change in 0.1 second is about 0.077 °C, which happens at the first moment of the touch event with the maximum temperature difference between set-up and finger. *Figure 11* shows an actual touch event data collected at ambient temperature around 20 °C. The temperature change in every 0.1 second is compared with the theoretical result.

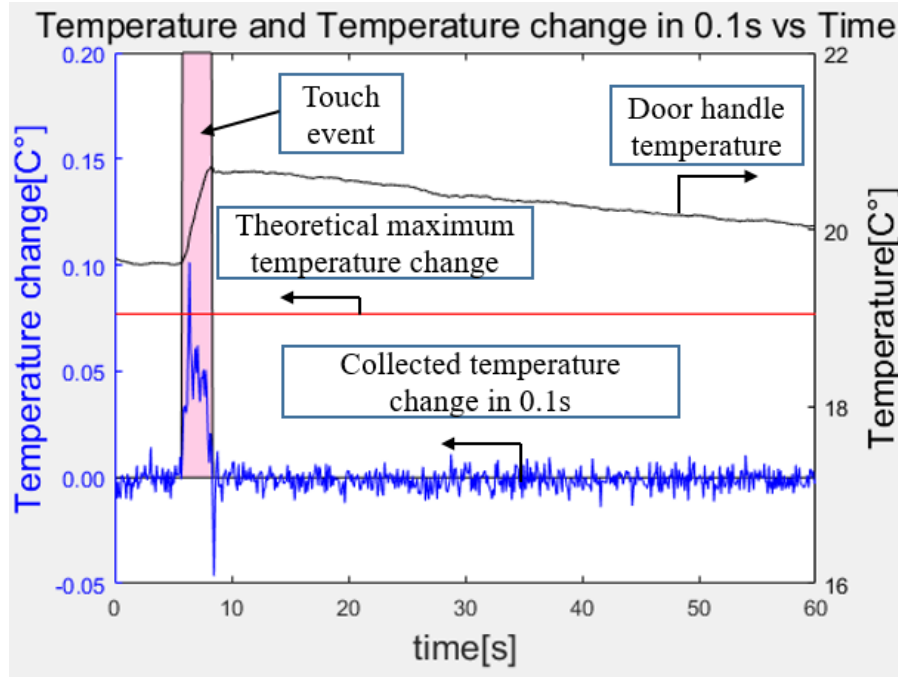


Figure 11: Comparison of theoretical maximum temperature change in 0.1 second and actual temperature and temperature change profile with experimental set-up.

The actual maximum temperature change in 0.1 second from this experiment set-up is 0.101 °C. Compared to the theoretical value of 0.077 °C, the collected data from the current set-up is close to that number. This validates that the current set-up is good for showing the temperature change of the door handle.

2.1.5. Results & Discussion

Discrete temperature data was collected using the temperature sensor. Touch detection methodology has been tested for both “false negatives” and “false positives”. 300 touch at different times from different days with different ambient temperatures were tested to check if there was any “false negatives”. And ensure no one touched the door handle for a period of two weeks (the temperature only changed with ambient temperature) to see if any “false positives” had been detected. A brief result is shown in Table 4

Table 4: Results of touch testing compared with the ground truth

	Actual touch event	Detected touch event
Touch test	300	296
No touch test	0	0

The methodology was able to detect more than 98.5 % of isolated touch events within the ambient temperature range of 10 to 25 °C. Figure 12 shows the temperature data and resulting touch event detection for a select minute containing two touch events.

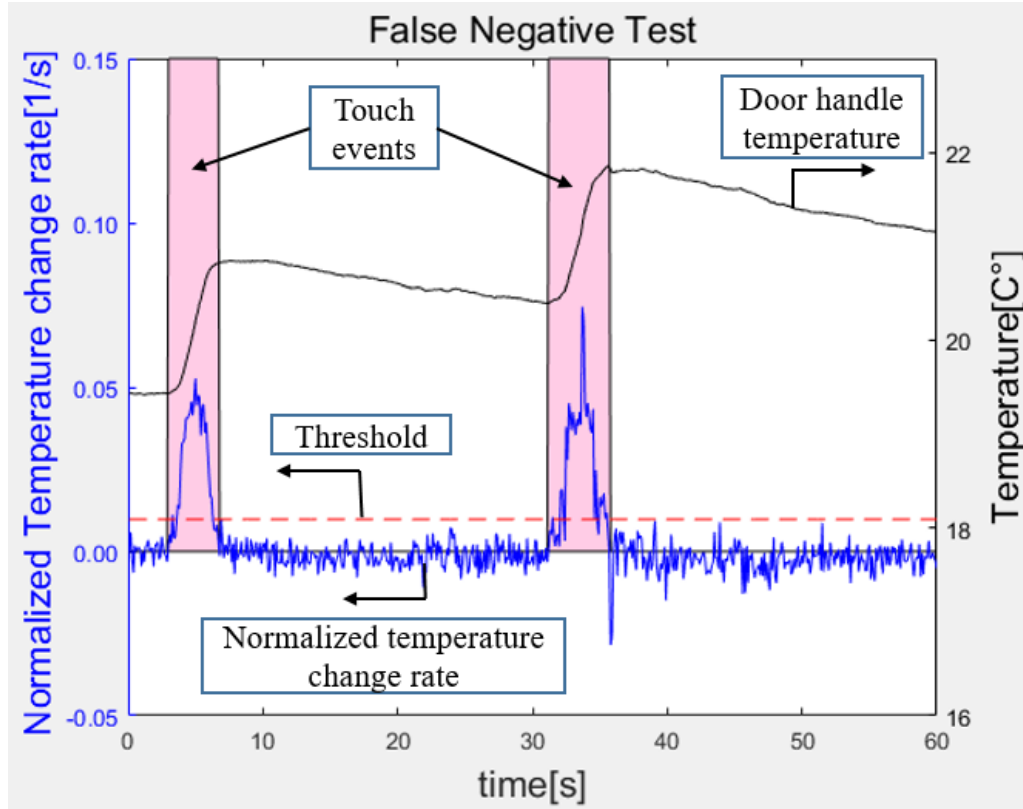


Figure 12: Demonstration of detected touch events by using normalized temperature change rate data and threshold.

The false positive test measured and recorded the temperature data for 3 hours long. And made sure no one touched the door handle during that time, with the temperature of set-up increased from 23 °C to 25 °C by natural convection. Figure 13 shows the temperature and normalized temperature change profile for this test, with the result of no “false positives” observed.

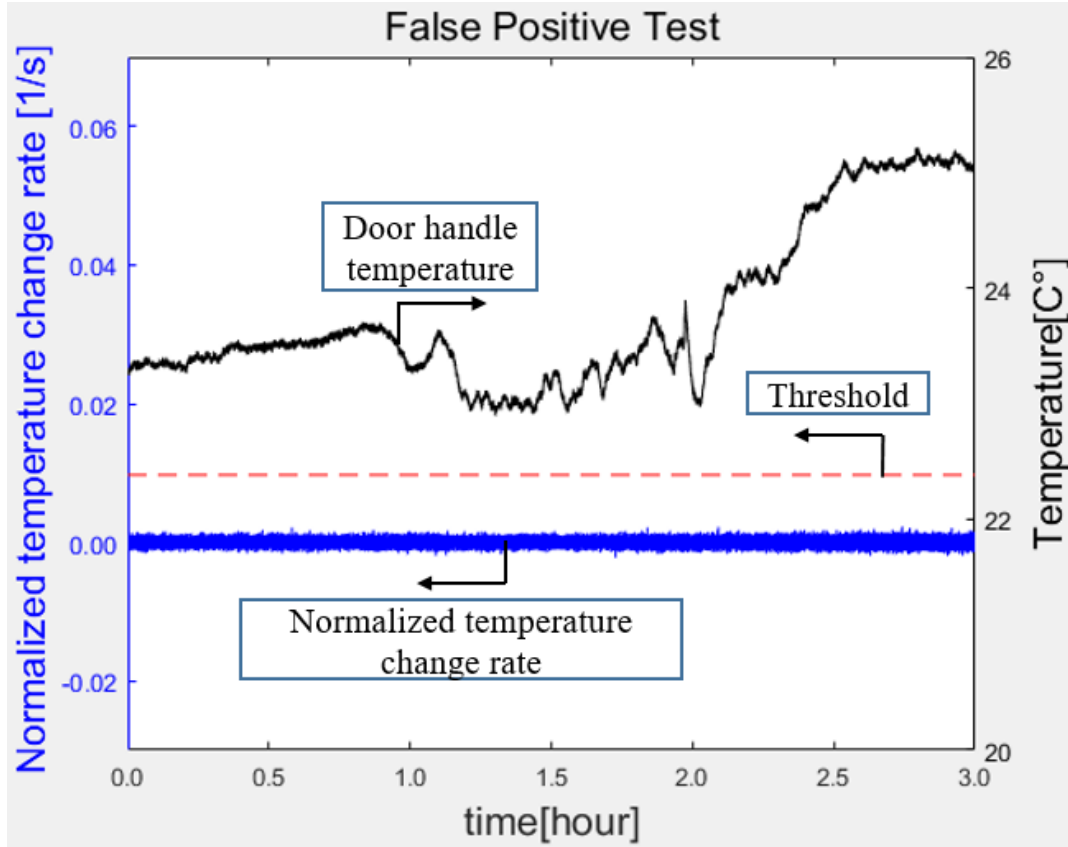


Figure 13: Demonstration of no misdetection during this 3 hours long test. Normalized temperature change rate keeps at the level about 0 when the temperature increased with the ambient temperature. During the process, normalized temperature change rate

2.2. Water Usage Detection

Due to the temperature difference of human's hand and the door handle, the event of a person touching a door knob/handle can be detected by measuring and recording the rate of temperature change. A similar concept can be used for detecting water usage by monitoring temperature of an inlet water pipe. Since water from ground is much colder than water that stays in pipelines, the pipe temperature will drop significantly when someone uses the water.

During a water usage event, the temperature change rate of the water pipe is dominated by the heat transfer with the cold water (from ground). The heat balance equation is expressed in Eq. (2.11).

$$mC_p \frac{dT}{dt} = hA(T_{W_ground} - T(t)) \quad (2.11)$$

where T_{W_ground} is the temperature of cold water from the ground, A is the inner area of the pipe, and h is the overall heat transfer coefficient. By integrating Eq. (2.11), the normalized temperature change rate can be derived, Eq. (2.12),

$$\frac{dT}{T_i - T_{W_ground}} = e^{-b\Delta t} \quad (2.12)$$

$$b = \frac{hA}{mC_p} \quad (2.13)$$

where m is the mass, and T_i is the initial temperature of the water pipe. In this work I assume T_{W_ground} is 10 °C.⁸³ The rest of temperature data processing, including moving average and threshold value updating, are the same as the door handle touch model. Table 5 summarize all the key parameters used in this model.

Table 5: List of the key parameters used in door handle touch detection model.

Water Usage Detection Model	
Moving average window	1 second
Groundwater temperature	10 °C
Minimal Activity Duration	1 second
Confidence level	95%

2.2.1. Experimental Set-up

An experiment was developed to validate water usage event detection. The test bed used for analysis consisted of tap water usage detection and a toilet flush detection. For each model, one K-type thermocouple was attached to the inlet of the water pipe to monitor the temperature change of the cold inlet. A schematic view of the apparatus for measuring the temperature of the inlet water pipe is shown in Figure 14. The temperature data of the inlet water pipe were measured and recorded with a NI Compact DAQ at 1 Hz. The uncertainty in temperature measurement is about ± 0.2 °C. All data by the temperature sensor was averaged by using the moving average method. Application of K-type thermocouple was suitable for experimentation due to a wide temperature range, high precision for quantitative measurements, and cost effective for practical deployment.

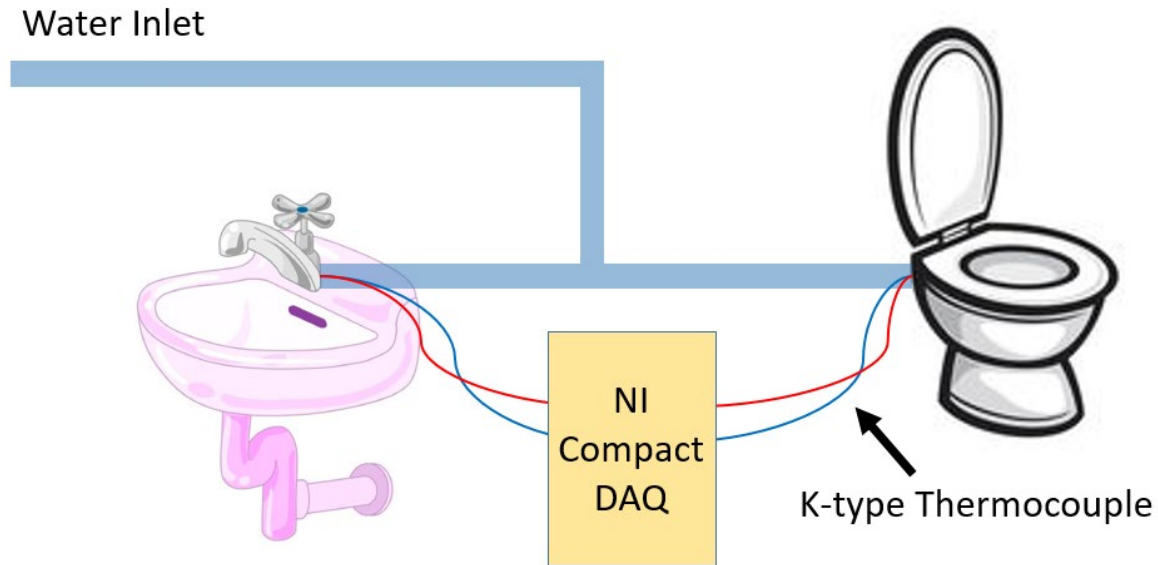


Figure 14: Schematic diagram of experimental set-up measuring temperature of this sensor with a K-type thermocouple.

To simulate home usage, tap and toilet was periodically drawn during the day time. Water draws were performed and recorded over the course of two months. There was at least 120 draws event during this period. Draw duration ranged from 5 seconds to 15 minutes to simulate different water usage event (wash hands/ flush toilet/ shower).

2.2.2. Results & Discussion

Discrete temperature data was collected and post-processed by LabView.⁸⁴ After comparing the result with a log file that recorded the actual water usage event, normalized temperature change rate based detection methodology was able to identify 100 % of isolated water usage events within two months. There are no falsely detected water usage events during the testing time. A brief result is shown in Table 6.

Table 6: Results of Toilet Water Usage and Tap Water Usage events compared with expected values

	Actual Water Usage Event	Detected Water Usage Event
Toilet Usage	45	45
Tap Usage	83	83

A single draw event is shown in Figure 15, showing normalized temperature change rate was accurately extracted for a water usage event as intended. All detected start times matched the actual start times within 2 seconds. For quick water usage events (less than five minutes), the detection model can retrieve the event's duration within a solution ± 2 seconds. Since the water inlet pipe will reach thermal equilibrium after five minutes, the current set-up is unable to detect the actual event's end time.

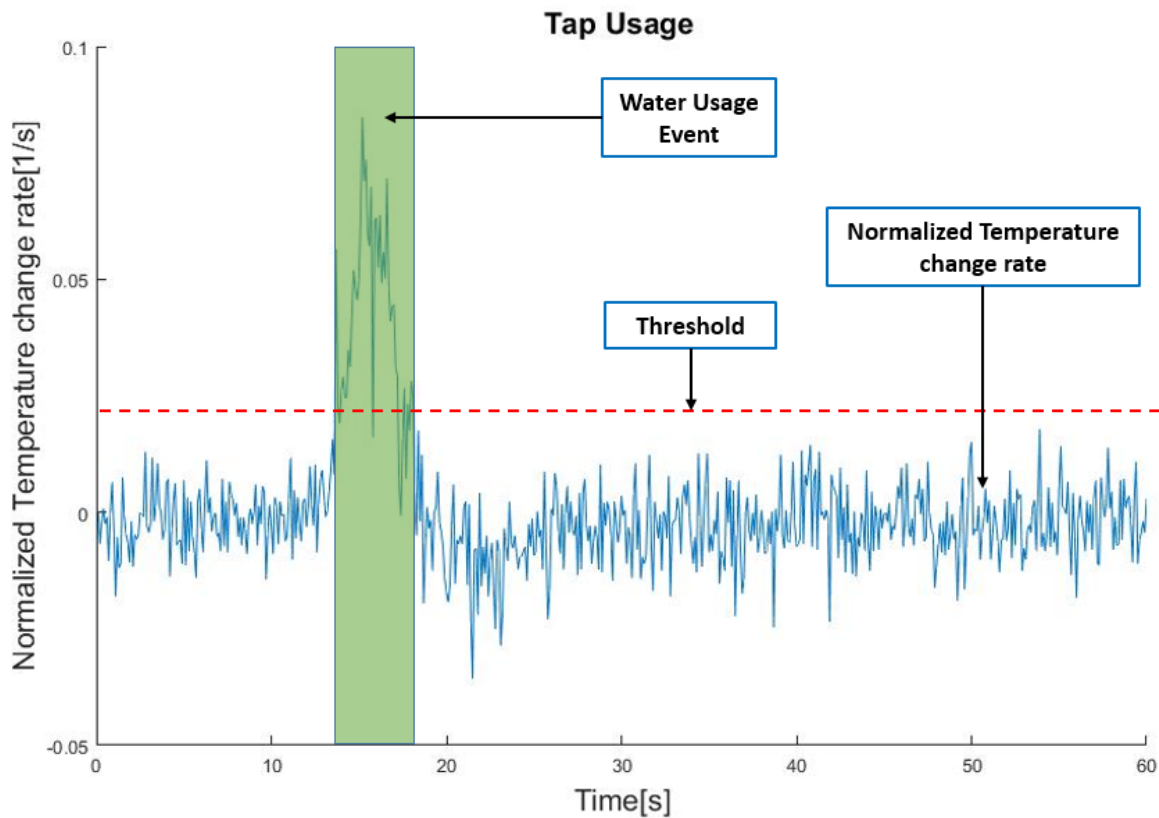


Figure 15: Demonstration of a single water draw event data in a selected minute, where the normalized temperature change rate increased rapidly and exceeded the threshold during the water usage event.

2.3. Passive Infrared Motion Sensor

Passive infrared motion sensor (PIR) measures infrared light radiating from objects in its view to detect movement of a warmer-than-background object in the surrounding areas. Since its powerful function and low-cost advantages, it has been adopted in many products. PIR sensors are generally used in security system to detect when someone is in the building when they shouldn't be.

However, PIR motion sensors can be falsely triggered by any hot source, including open heating elements, incandescent lightbulbs, and convective heat currents. Since PIR motion sensor can only detect a fixed field-of-view, it requires a technician to fine-tune the set-up and wiring design based on the house floorplan. After installation, any move of furniture and equipment with respect to the sensor needs to be tested. Low accuracy and hard-to-install hinder the application of PIR motion sensors in residential building energy management systems.

This thesis suggested installing a PIR motion sensor at the top of the inside door frame of the entrance to address accuracy and installation concerns. The sensor is face to the ground of the door area. Since this area is reserved for the door opening, people will not place any heating device or furniture in this area. Figure 16 is a diagrammatic representation of the sensor installing position and the field-of-view. The PIR motion sensor was connected to a system-on-a-chip, Beaglebone Black.⁸⁵ A java code was used to collect the motion data at 1Hz via a general-purposes input/output port on Beaglebone Black. With this set-up, the PIR motion sensor can only detect the movement near the door area.

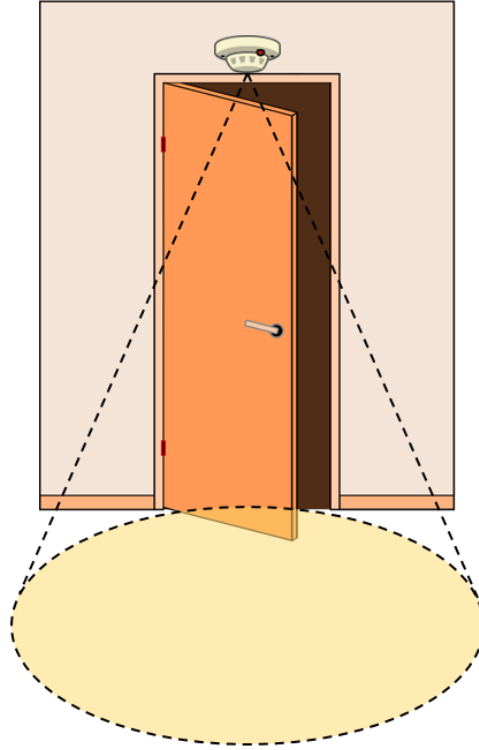


Figure 16: PIR motion sensor is installed on the top of the inside door frame to detect any movement near the door area.

This chapter discussed three low-cost and non-intrusive detection models which recognize human activities from temperature and motion data. Automated algorithms were developed based on thermal conservations to detect when door handle touch and water usage. The moving average method was applied to the raw temperature data in order to smooth out the fluctuation happens one second. Then the temperature change was calculated and normalized with the ambient temperature, sampling rate and hand/ water pipe temperature. The normalized temperature change rate, $\Delta \dot{T}$, was compared with a threshold value, which is calculated based on the mean and standard deviation of maximum value of $\Delta \dot{T}$ collected in all previous human activities data. If $\Delta \dot{T}$

exceeded the threshold value for more than one second, the sensor was considered to detect a new human activity. The maximum value of $\Delta \hat{T}$ was then recorded and used to update a new threshold value.

Based on the result, the door handle touch detection model and water usage detection model can detect 98% and 100% of the human activities respect without any false triggering. The uncertainty of the conventional PIR sensor used in this work is 1 %. Since the accuracy of human activity detection models is relevant high ($> 98\%$), the rest of the thesis does not consider the 1% chance of incorrect activity detection. Moreover, this thesis suggested installing a PIR motion sensor at the top of the inside door frame to detect human motion near the door.

Chapter 3. Occupancy Detection via Machine Learning

The previous proposed sensors can effectively detect human activities that are highly correlated with human occupancy. However, human activities, such as water usage, are not always related to occupancy information, which may mislead occupancy information in some cases. Figure 17 shows the time frame of two selected examples. Case 1 shows someone comes back home to grab his cellphone and leave the building in one minute, so there is no occupant in the building afterward. Case 2 reveals two occupants come back home in a short period. As a result, it will be two occupants in the building. Both cases contain two outside door handle touch activities and two door-area motion activities. If we rely on one single human activity sensor, neither door handle touch nor door-area motion activities can differentiate these two cases by themselves. Hence, I proposed a data fusion scheme to combine all the human activities happens in a short period and predict the occupancy change information based on that via ML algorithms.

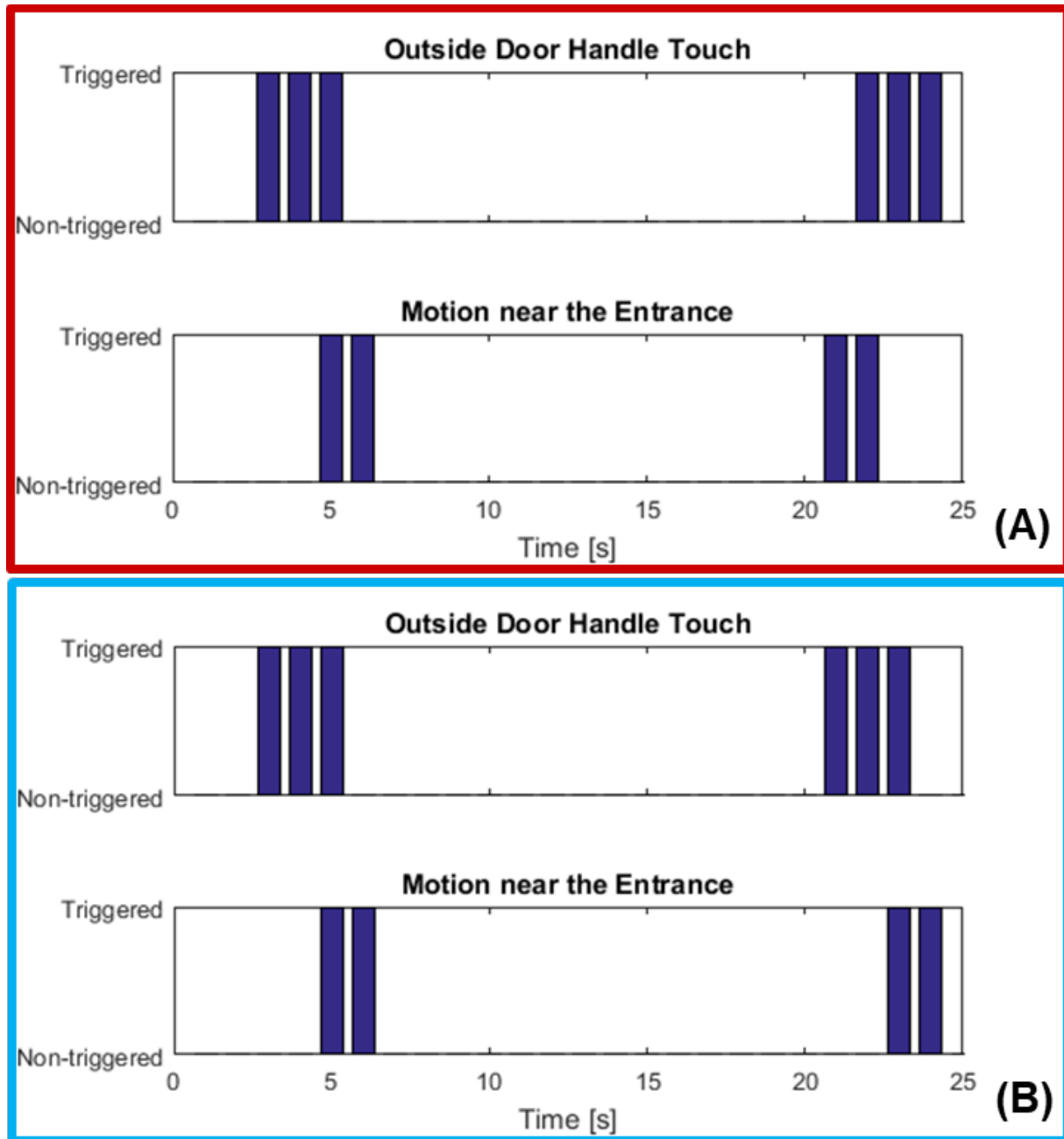


Figure 17: Description of two selected cases: (A) grab a cell phone and (B) two users back to home in sequence. Both cases contain two outside door handle touch activities and two door area motion activities.

3.1. HLA Based Simultaneous Data Collection

Integrating different types of human activity detection models will trend to more accurate and detailed occupancy information. But these sensors are installed at various locations in the house, using different programs or standards to record and transfer data. This will be a huge challenge for the data collection to gather information from each of these diverse detection models. Since the sequence pattern of human activities is essential, the data collection needs to handle the time sensitivity of data.

This thesis proposes the development of a distributed sensor collection scheme of heterogeneous human activity detection models based on High Level Architecture (HLA).⁸⁶⁻⁸⁸ HLA is a general-purpose architecture initially developed by the U.S. Department of Defense in 1998. The principal idea of HLA is to separate the functionality of each entities using a general proposal infrastructure, called Runtime Infrastructure (RTI).⁸⁹ RTI provides synchronous data exchange while accurately controlling time step progression between entities. Since each entity only need to communicate with RTI, Figure 18 shows implementation of the RTI can vastly improve interoperability between entities.

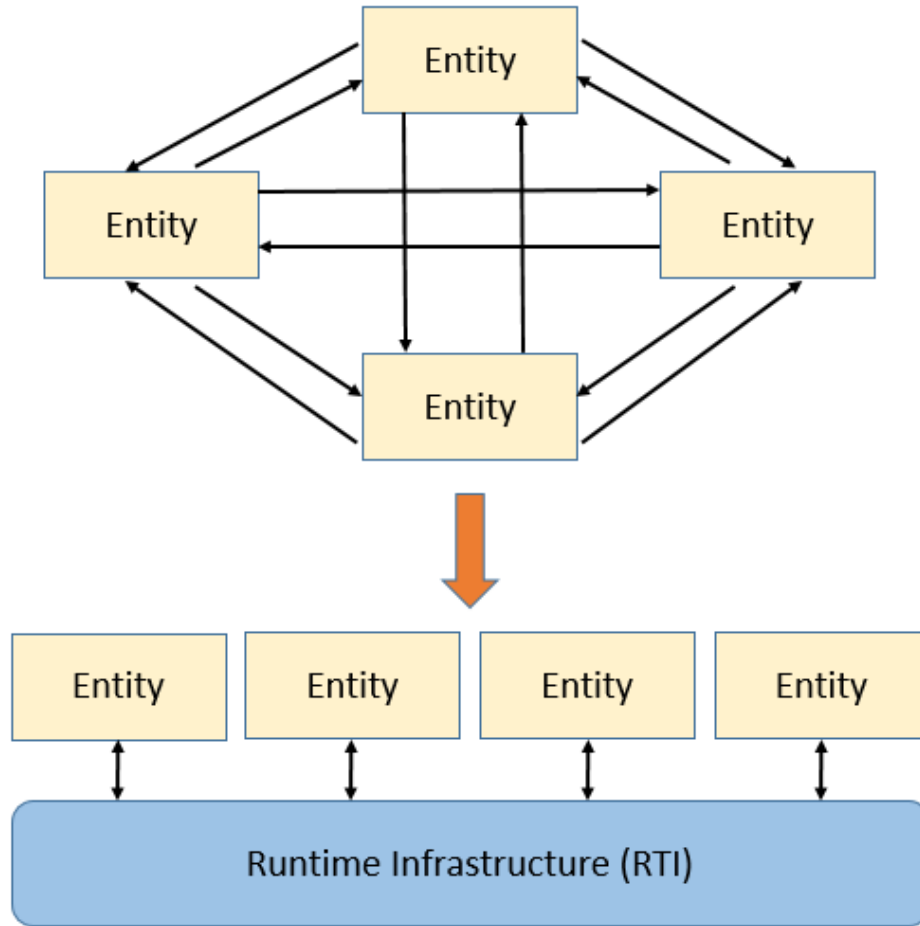


Figure 18: Instead of having individual connectivity between entities, the RTI environment can more simply connect federates for co-simulations.

Due to the high interoperability and low complex, HLA is widely used to aggregate simulations distributed across heterogeneous hardware and software platforms. Lim and Kim⁹⁰ developed a HLA based framework, HDEVSimHLA, for interoperating the DEVS⁹¹ models and the MATLAB/Simulink⁹² models. Albagli et al.⁹³ proposed an integration of simulators MATLAB/Simulink, Omnet++⁹⁴ and JADE⁹⁵ to simulate smart grid applications. A similar set-up is used in the sensor data collection scheme to synchronize data from different human activity detection models. At each time step, human activity detection models send the collected data to the HLA independently.

After receiving all the data, HLA forms the data, adds a timestamp, and sends it to the database.

Such a connection can be easily implemented for PIR motion sensor since the data collection tool on BeagleBone Black is just a java code that can be readily modified to compact with RTI. However, for touch detection and water usage detection models, LabView is utilized to collect raw temperature data. LabView and RTI are not currently compatible. They have different notions of time management, and HLA does not support data exchange using LabView. In order to communicate with LabView, a bi-directional message forwarder is devised via Transmission Control Protocol/Internet Protocol (TCP/IP). TCP/IP⁹⁶ is a common communications protocol used to interconnect network devices on the internet. It provides stable client-host model of communication that identify how data should be broken into packets, addressed, transmitted, routed, and received at the destination. TCP/IP is designed to make networks reliable since it could recover automatically from the failure of unexpected disconnection.

Figure 19 shows the flowchart of data collection using LabView. At each time step, LabView first collects raw temperature data. Next, human activity detection models convert the temperature data into human activity data. After receiving a Start Token from HLA, LabView sends all the human activity data with an end token back to HLA. Then goes to the next timestep. By sending and receiving the tokens, LabView is able to synchronize with other entities, so the actual sequence of activities can be collected.

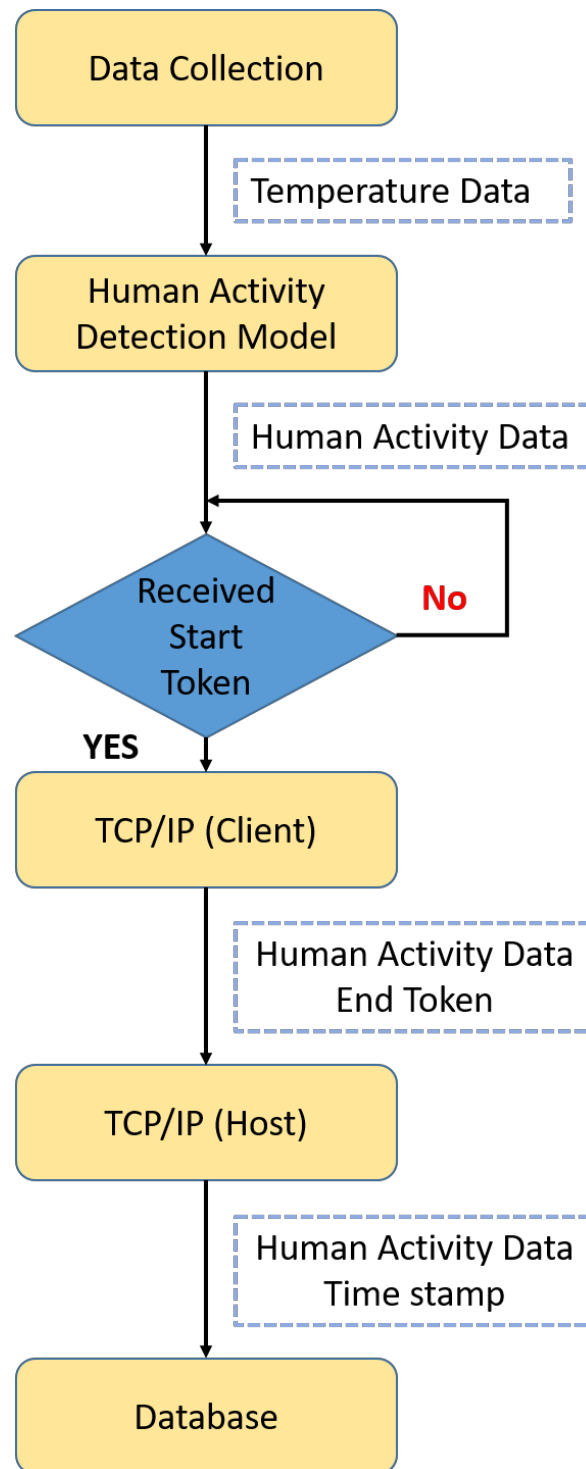


Figure 19: Flowchart representing how data is collected, processed and sent to the database via TCP/IP.

The proposed HLA-based data collection scheme contains five entities. Touch and water usage data are collected with two separate computers at the entrance area and bathroom. A BeagleBone Black on the door frame is used to collect PIR motion data. The actual occupancy information (ground truth) is logged by another BeagleBone Black located on the wall of the entrance. HLA is running on another computer, which is also used to gather and record human activities data into the database. Figure 20 shows the schematic diagram of the proposed HLA based simultaneous data collection scheme.

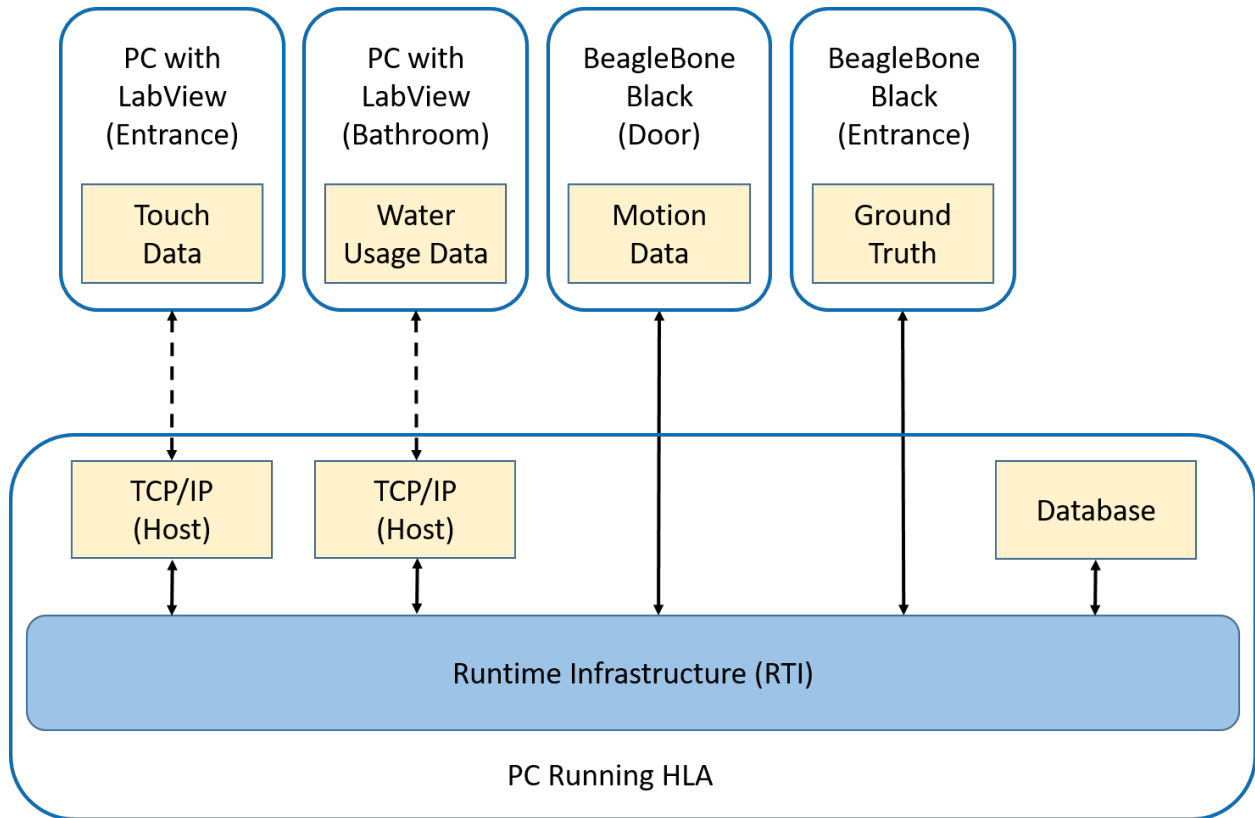


Figure 20: Schematic diagram of the proposed HLA based simultaneous data collection scheme. Two PC entities connect via TCP/IP forwarder. Two BeagleBone Black entities remotely communicate with RTI. Database entity is operating on the same computer that runs HLA.

3.2. Machine Learning Algorithms

Occupancy Classification and regression are two major prediction problems that ML models are designed to handle. Classification is the process of categorizing data point into multiple categories. Regression aims to predict a data value based on a function defined by the available data points. In this paper, occupancy detection is treated as a classification problem rather than a regression problem for the following reasons: (1) the number of occupants in a residential house does not vary as much as commercial buildings; (2) the energy consumption in a residential house primarily depends more on the state of occupancy rather than the number of occupants.

The four classification models, Random Forest; Decision Tree; K-Nearest Neighbor; and Support Vector Machine were used for comparison in this work, because these algorithms represent the most popular ML algorithms used in different application scenarios. These algorithms were briefly described in this chapter 3.2.

3.2.1. Decision Tree

Decision Tree algorithm uses a series binary question to classify data. The algorithm determines the most important attributes and questions of the training data in a way to reduce uncertainty based on information entropy or Gini index. In this work, the Gini Index was chosen as the standard, which reflects the probability of a particular input being falsely classified:

$$Gini = 1 - \sum_{i=1}^n p_i^2 \quad (3.1)$$

where p_i is the probability of one data point (object) being categorized into a class i . Decision Tree algorithm starts with the full data set called the root. The data set is broken down into two subsets by asking the binary question of the feature with the least Gini index. The model keeps splitting the data set into smaller subsets until all data points in the subset are labelled with the same classification, called a leaf. The flow from the root to the leaves can be treated as a classification rule. All future data samples that follow the same rule can be classified into the same class, the class of the leaf. Applications of the Decision Tree algorithm often use questions highly specific to a single data set, so the model cannot easily be transferrable to new data (overfitting issue).

3.2.2. Random Forests

Instead of one decision tree, random forests have multiple decision trees, which were constructed from randomly chosen subsets of data. Each tree is made with different input features, so that each decision tree can lead to a different decision for classification. Instead of relying on a decision from a single decision tree, the final decision is made based on the majority vote of different trees in the forest. Because Random Forests intentionally limit the number of features to train within each single tree model, it has less overfitting problems than the conventional decision tree. However, the random forests may require more input features and data for training.

3.2.3. K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm is widely adopted in classification problems in industry because it is easy to understand and interpret. KNN assumes that similar things belong to the same class based on the distance-based, non-linear, and non-parametric method. The algorithm classifies a new data point (t) by evaluating the information distance (d_i) between t and labelled instances (x_i) in an existing data set:

$$d_i = \sqrt{(t - x_i)^2} \quad (3.2)$$

A majority vote will be performed using the K instances with the smallest distance, and the new data point will have the label that wins the vote. The training process will determine how many neighbors (K) need to be considered for the voting process. However, the training time and memory requirements are high, and the prediction process might be slow with big training data.

3.2.4. Support Vector Machine

Support Vector Machine (SVM) is another machine learning algorithm based on minimizing the risk, which is first proposed by Boser in 1992.⁹⁷ The main objective of SVM is to construct a boundary that best separates a dataset into different classes. Support vectors are the data points nearest to the boundary, which are considered as the most critical elements to determine the boundary. SVM sets the boundary in a way that the distance from the boundary to each class is maximized, so that future data can be classified with more confidence. Due to its effectiveness in handling large

number of features, a few studies explored SVM model to forecast occupancy information.^{98,99} The results showed that SVM-based models achieved more than 80 % accuracy in most scenarios.

3.3. Experimental Set-up

The experiments were conducted in the living lab, Solar House, at the Santa Clara University. The area of the room is about 62 m², including a living room, a kitchen, a bedroom, and a bathroom. The house has a small occupancy change (1~5 occupants). During the experiments, the only assumption is that the occupants will close the door after they enter or leave the house. The schematic floor plan and sensor distribution are shown in Figure 21.

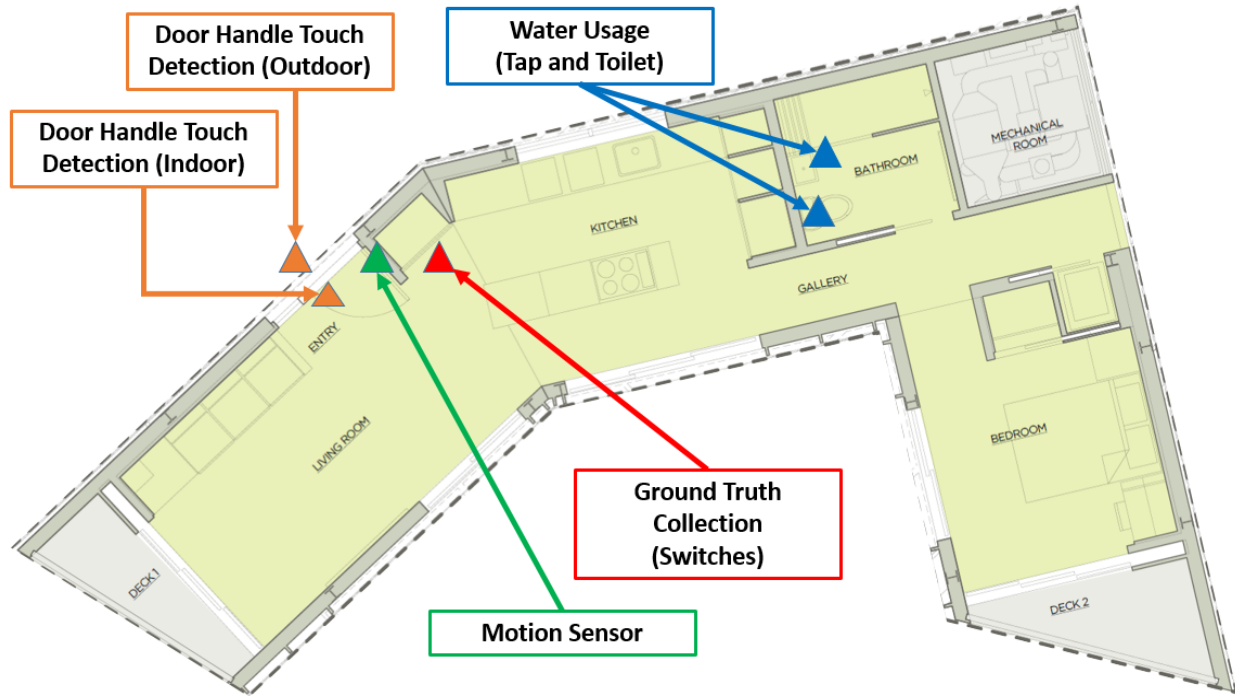


Figure 21: Schematic floor plan of experimental set-up detecting occupancy information in a living lab with multiple types of sensors and 5 switches for ground truth collection.

Two door handles touch detection sensors (indoor/outdoor) was installed at the entrance of the house. Two water usage detection sensors were attached to the water supply pipes of the tap and toilet in the restroom. All raw environmental data were fed into the human activities layer to detect activities (Indoor Handle Touch/ Outdoor Handle Touch/ Tap Usage/ Toilet Usage). Transient and steady-state normalized heat transfer models were developed for door handles/ water pipes, human hand/ water, and environment to consider the variance of the ambient temperature and contact quality. A passive infrared sensor (PIR) was installed on the top of the door frame (indoor), which can directly detect if someone passes the entrance area. Five wall switches, each one representing one person, collected the ground truth of the occupancy information, which was used as the label for the dataset. A camera was

also employed to detect sensor malfunction or mistakes in the ground truth collection. All the temperature data were measured and recorded by LabView⁸⁴ with an NI Compact DAQ at 1 Hz. The motion data and ground truth were collected with a system-on-a-chip, Beaglebone Black,⁸⁵ also at 1 Hz.

3.4. Data Processing

The sampling rate of the raw environmental data collection was empirically optimized to be 1 Hz in order to detect corresponding human activities. The raw data were first fed into the human activities detection layer to recognize if an activity happens at this time step.

The occupancy information depends on the total number and the sequence of human activities that occur in a short period. The sequence of activities used to estimate occupancy are summarized into an event window. The end of an event window is determined when no further activities are detected for more than 30 seconds after the last detected activity. Figure 22 shows the duration distribution of 487 events collected in this work with a minimum at 1 s and a maximum at 316 s.

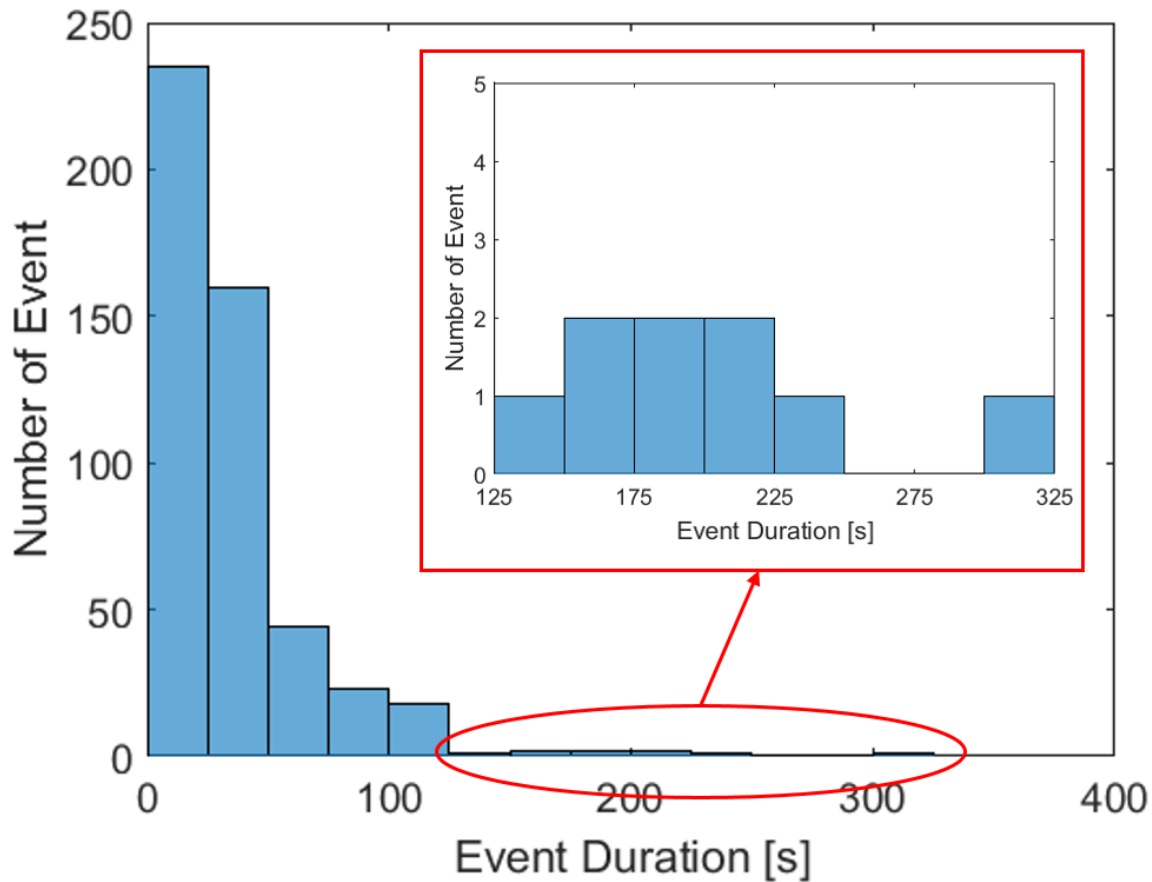


Figure 22: Histogram of event duration of 489 events collected in this work.

Depending on how activities occur, the length of event windows may change even for the same event. Some entering events are associated with indoor door handle touch, and some are immediately followed by water use. As such, the duration of each event will differ. Typical ML algorithms can only accept training data with fixed sizes. Hence, a fixed format was devised to summarize the event regardless of its total duration.

Figure 23 shows an example of an event window that happened at 8:37:09 on 10/23/2019. The event window started at 8:37:09 and ended at 8:37:19, which means no activity occurred in the 30 s period before 8:37:08 or after 8:37:20. The five

columns, including Indoor Handle, Outdoor Handle, Tap, Toilet, and PIR indicate 5 corresponding human activities, inside door handle touch; outside door handle touch; tap usage; toilet usage; and motion near the door area. The actual occupancy information was collected by five wall switches, each one corresponding to one occupant working in the building. There are three labels of the ground truth: "1" means someone enters the house, "-1" means someone leaves the house, and "0" means no change of occupancy information. Figure 23 shows an example of an entering event. The user touched the outside door handle to open the door at 8:37:09 and triggered the motion sensor near the door area three times at 8:37:09, 8:37:14, and 8:37:19 respectively.

Event Window							
Time	Event Relative Time	Indoor Handle	Outdoor Handle	Tap	Toilet	PIR	Label
8:37:09	1	0	1	0	0	1	"1": entering event
8:37:10	2	0	1	0	0	0	
8:37:11	3	0	0	0	0	0	
8:37:12	4	0	0	0	0	0	
8:37:13	5	0	0	0	0	0	
8:37:14	6	0	0	0	0	1	
8:37:15	7	0	0	0	0	0	
8:37:16	8	0	0	0	0	0	
8:37:17	9	0	0	0	0	0	
8:37:18	10	0	0	0	0	0	
8:37:19	11	0	0	0	0	1	
Relative Time Event Sequence							
Event Date & Time	Indoor Handle			Outdoor Handle			Label
	1st	2nd	3rd	1st	2nd	3rd	
10/23/2019 8:37:09 AM	0	0	0	1	0	0	"1": entering event
	Tap			Toilet			
	1st	2nd	3rd	1st	2nd	3rd	
	0	0	0	0	0	0	
	PIR						
	1st	2nd	3rd	Last			
	1	6	11	11			

Figure 23: An example of an entering event window: the event started with an outside door handle touch activity followed by motion near the door area. The data from this event window can be converted into a fixed-length format.

The lower table in Figure 23 shows how to convert the example event into a fixed-length format, relative time event sequence. For each activity, the relative time event sequence only records the relative time in the event window of first, second, and third nonconsecutive trigger time of each sensor. Since the motion sensor can be triggered frequently, the last trigger time is also kept. All the event data are converted into a fixed-length format, which is ready to be used in ML models.

Figure 24 illustrates the overall schematic diagram of data processing. First, the raw environmental data (temperature and motion) is converted into human activity data by using the white-box models introduced in Chapter 2. Second, multiple event windows are extracted to describe the human activities happens in a short period. Based on how activities occur, the duration of events is different. As a result, the size of data matrix size is varying. Then all the event window data is summarized into a fixed-length format, relative time event sequence format, regardless of its actual duration. After inputting into ML algorithm, and the ML model will make a final prediction from “entering event”, “leaving event” and “no change event”.

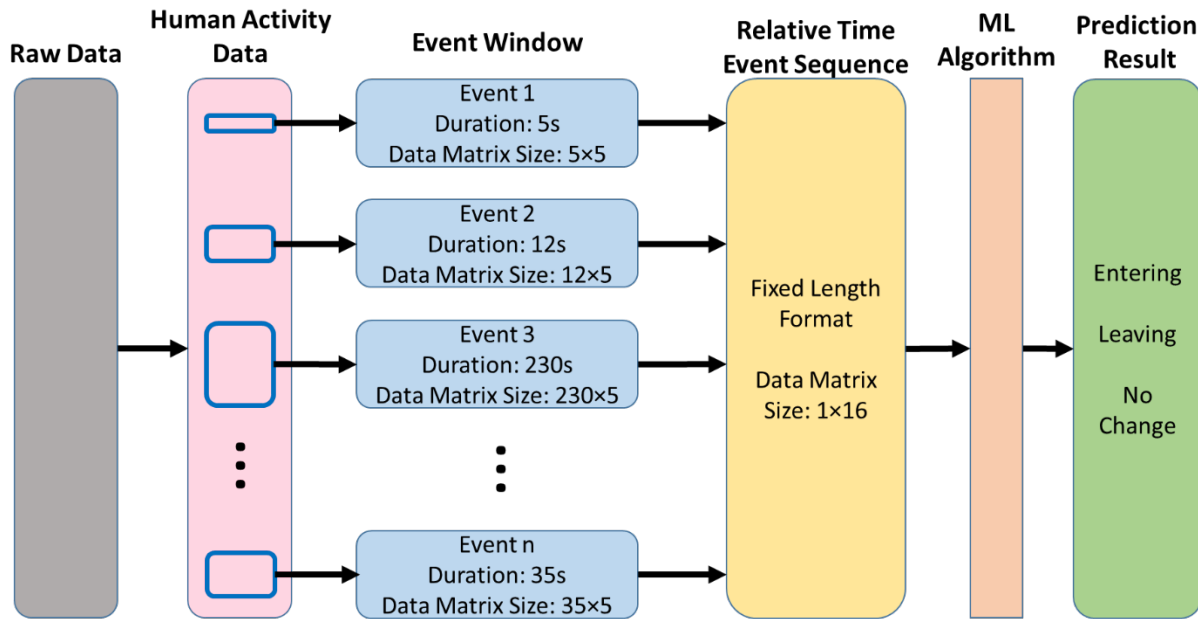


Figure 24: Schematic diagram of data processing from the variable-size event window to the final predicted occupancy information.

3.5. Results & Discussion

The proposed scheme contains two layers of models. The accuracy of the lower layer (human activities detection) is very high (>95 %), which has been tested in Section 2.1. This section focuses on showing the result of the upper layer (ML-based classifier).

Powers¹⁰⁰ introduced several criteria to analyze the validity of the classifiers. All the criteria are based on the True Positive (TP), True Negative (TN), False Negative (FN), and False Positive (FP). Among them, TP and TN represent the correct classification if the test data belong to the correct label class. FN and FP represent the incorrect prediction if the entry does not belong to the negative or positive classes, respectively.

Accuracy measures the percentage of entries that were correctly classified Eq. (3.3).

Recall measures the rate of TP entries to all correct predicted entries Eq. (3.4).

Precision measures the fraction of correct positive predictions to the total predicted positives Eq. (3.5). The F1-score is a technique that measures the discrimination of classes through recall and precision, which is equal to harmonic average of recall and precision Eq. (3.6).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.3)$$

$$Recall = \frac{TP}{TP + FN} \quad (3.4)$$

$$Precision = \frac{TP}{TP + FP} \quad (3.5)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3.6)$$

The case proposed in this paper is not a simple binary classification problem. The data contains four different classes: someone entering, someone leaving, no change with someone in the house, and no change without anyone in the house. The Macro-average method¹⁰¹ suggests taking the average recall and precision of each class, shown in Eq. (3.7):

$$\bar{f} = (f_1 + f_2 + \dots + f_n) / n \quad (3.7)$$

so recall/precision of the proposed model is equal to the average of the recall/precision of entering, leaving and no change class. The overall F1-score is calculated based on the averaged values of recall and precision.

The data collection for this research occurred over a two-months period, and 489 events were detected. I organized all the data samples according to the time when they were collected. The first 80 % of the data, which were collected during the beginning 80 % of the time, were used for training and the remaining 20 % of the data were used for testing. A Python based machine learning library, scikit-learn,¹⁰² was used for data analysis in this work. The four classification models, Decision Tree (criterion = gini); Random Forest (number of estimators = 100, criterion = gini); K-

nearest Neighbors (number of neighbors = 5, metric = minkowski); and Support Vector Machine (c = 1, kernel = radial basis function, gamma = scale) were trained and tested with and without the human activities layer. Table 7 summarizes the detailed results of the upper layer (ML-based classifier) obtained in this research. Four criteria (accuracy, precision, recall, and F1-score) were calculated to compare the validity of the classifiers by using Eq. (3.3) – Eq. (3.6).

Table 7: Comparison of the performance of applying Decision Tree, Random Forest, K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) algorithms to the data with and without the Lower Layer (Human Activities Layer).

	Without Human Activities Layer			
	Decision Tree	Random Forest	K-Nearest Neighbors	Support Vector Machine
Accuracy	0.86	0.86	0.86	0.87
Precision	0.78	0.80	0.71	0.62
Recall	0.51	0.51	0.52	0.51
F1-score	0.62	0.63	0.60	0.56
	With Human Activities Layer			
	Decision Tree	Random Forest	K-Nearest Neighbors	Support Vector Machine
Accuracy	0.96	0.99	0.95	0.90
Precision	0.93	0.98	0.94	0.96
Recall	0.93	0.98	0.91	0.79
F1-score	0.93	0.98	0.93	0.87

Out of the four criteria, F1-score is the most critical criterion, since false positives and false negatives are more crucial in an occupancy detection model. False negatives are related to turning off the HVAC system when someone in the house, which may

decrease users' comfort and cause some health issues. False positives can reduce energy conservation by operating the HVAC during the unoccupied period.

When the human activities layer was not implemented, none of the four algorithms had high enough F1 scores to make a reasonable prediction. Conventional ML algorithms require a sufficient density of sensors to ensure that the collected data can accurately reflect occupancy information in the house. Otherwise, the algorithm may not be able to make a proper classification. For example, if the approach only installs a few temperature sensors in the bathroom, occupancy information for the other rooms or the entire building cannot be determined from the data. Due to the diverse nature of residential buildings (e.g., floorplan, material, location, and orientation), the number/type/location of the sensors can hardly be generalized. All these ML algorithms have been proven capable of detecting occupancy with a higher number of sensors in previous researches.^{37–41,43,44,46,48–50,97–99,103,104} The sensor density in this work is not sufficient to provide information for these standard ML approaches. Moreover, the raw data is highly unbalanced: unoccupied data are much more than occupied data, which tends to have more FN cases. In order to make it work with the existing set-up, more sensors would need to be installed and the data would require additional preprocessing which requires expertise in sensor distribution and model training, accompanied by the high cost of the system and complex installation.

Incorporating the human activities layer based on domain knowledge increased the performance of all classifiers. The overall accuracy had slightly increased by about 10 %. The F1-score was dramatically improved between 31 % to 35 %. Among these

four techniques, the Random Forest algorithm yielded the best accuracy and F1-score (both > 95 %). Table 8 demonstrates the detailed confusion matrix of the Random Forest algorithm. As we can see, the model effectively classified all events in the one-week long testing data, except one. One leaving event has been falsely classified as an entering event. Table 9 shows more detailed information on that misclassified event. In this event, the occupant touched the indoor handle, opened the door at the time step 4, and still stayed at the entrance area for over 30 seconds before leaving. This unusual event is similar to an occupant opening the door for someone else outside the house. As such, the event was misclassified as an entering event.

Table 8: Demonstration of the confusion matrix of Random Forest models. The matrix shows there is only one misclassified event in the testing data set.

Confusion Matrix		Predicted		
		Leaving	Entering	No change
Actual	Leaving	15	1	0
	Entering	0	16	0
	No change	0	0	65

Table 9: Illustration of the detailed information of the misclassified case.

Relative Time Event Sequence							
Event Date & Time	Indoor Handle			Outdoor Handle			Ground Truth
	1st	2nd	3rd	1st	2nd	3rd	
12/10/2019 12:54:41 AM	4	0	0	0	0	0	"-1": leaving event
	Tap			Toilet			Prediction
	1st	2nd	3rd	1st	2nd	3rd	
	0	0	0	0	0	0	
	PIR						
	1st	2nd	3rd	Last			
	1	5	9	37			"1": entering event

The Decision tree algorithm, although achieved relatively high accuracy and F1-score, can easily overfit to noises with high dimensional data. KNN and SVM models yielded worse performance because both models make their classification decisions based on measuring the distances among data samples, which actually represents the relevant time offset among sensor events in this application. However, such distances may be misleading in some cases. For example, the distance from a simplest entering event to the simplest leaving event (only trigger outdoor/indoor handle touch once at relative time 1) is smaller than the distance to a lingering entry event (trigger the outdoor handle touch multiple times in the event). The confusion matrices of Decision Tree, KNN and SVM are showed in the Appendix H.

Previous research that had more than enough sensors installed to ensure the data collected was able to reflect occupancy information. They tried different combinations of the sensors and ML algorithms to choose the model with the highest accuracy. However, this method increased the number of sensors, and thereby the cost of the whole system.

Figure 25 depicts the detection areas and the number of sensors of the approach described in this paper compared against other research. Among the prior works, several^[38,39,41,46,103] utilized 4 to 9 sensors in a single-person office/cubicle ($< 20 \text{ m}^2$), two^{99,104} employed 10 to 15 sensors in a multi-person office ($\sim 40 \text{ m}^2$), and one⁴⁴ used 24 sensors to estimate the occupancy information in a 100 m^2 lab. Based on the trend line, a 62 m^2 apartment would have required fifteen or more sensors for occupancy detection.

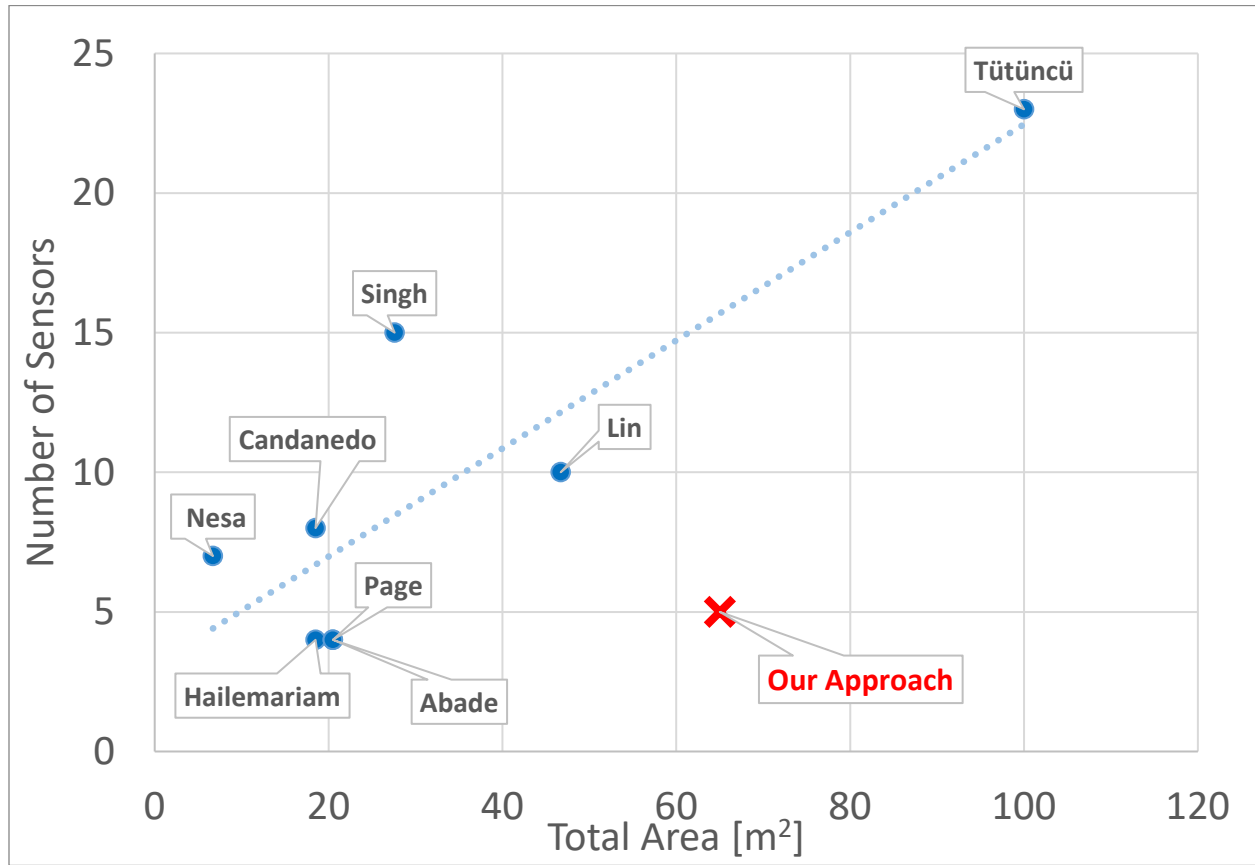


Figure 25: Comparison of the total detection areas and the number of sensors used by different research efforts in residential homes. All models used in these researches were able to provide valid occupancy information, with an error of misclassification less than 5 %.^{38,39,41,44,46,99,103,104}

Incorporating domain knowledge is an effective way to help machine learning models simplify the choice of representative features as well as reduce the number of sensors. Domain knowledge, in this context, is the understanding of human behavior in a residential home. The occupancy detection approach based on understanding of general human behavior only requires 5 sensors for a 62 m² living lab depicted as a red cross mark in Figure 25.

To further explore the potential of reducing the number of sensors, the weight of each activity is depicted in Figure 26. Since tap usage and toilet usage have the least significance (both less than 1 %), these two activities were removed and the Random Forest model was trained only with door handle touch and motion near the door area activities.

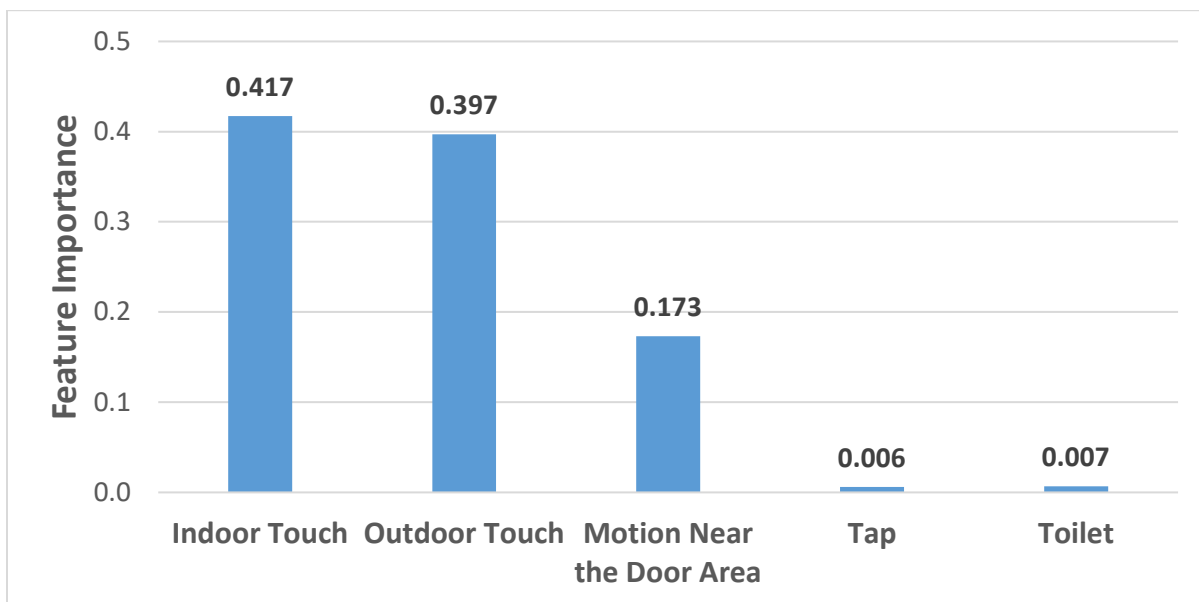


Figure 26: Demonstration of the importance of five human activities (door handle touches, motion near the door area, and water usages) in determining the occupancy prediction by Random Forest algorithm.

Table 10 shows the detailed results of the Random Forest model with and without water usage activities. After removing tap usage and toilet usage, the same level of accuracy and F1-score have been observed, which means the approach requires only three sensors to make an effective occupancy prediction in a 62 m² living lab.

Table 10: Comparison of the validity of Random Forest Models with and without the water usage human activities.

	Random Forest	
	w/ water usage	w/o water usage
Accuracy	0.98	0.98
Precision	0.98	0.97
Recall	0.98	0.97
F1-score	0.98	0.97

In this chapter, a solution for occupancy detection with a limited number of non-intrusive environment sensors was proposed and tested. Four machine learning-based classification algorithms (Decision Tree, Random Forest, K-Nearest neighbors, and Support Vector Machine) are then used to predict occupancy information with and without the human activities detection layer. The results showed that incorporating human activities models increased the performance of all classifiers. The overall accuracy had slightly increased by about 10 %, and the F1-score improved between 31 % to 35 %. Among these four algorithms, the Random Forest algorithm yielded the best accuracy and F1-score (both > 95 %). Moreover, the number of sensors can be

further decreased to three without risking the validity of occupancy prediction. In summary, incorporating human activities models can improve the performance of machine learning-based classifiers with a limited number of low-cost and non-intrusive sensors. Since human activities, like door open or water usage, are highly correlated with occupancy information regardless of the floorplan, material, location, and sensor orientation, such an approach can be readily transferrable to any residential building.

Chapter 4. Energy Saving Impact of Occupancy Information

Occupancy information can lead to a huge potential for energy saving in residential buildings. Whether it is worth investing such occupancy detection system is always the first question in people's minds. Michael¹⁰⁵ evaluates the payback analysis of each smart home components, including photovoltaic panels, wind turbines combined heating and power, energy storage systems, water heaters, electric vehicles, HVAC, and solar collectors. The result showed that the payback periods of most of the smart home components are higher than ten years. Since some smart home components have life-spans less than ten years, it is hard to convince customers to adopt the smart home system with such a long payback period.

This chapter assesses energy saving impact and economic benefits of occupancy information driven thermostats in a residential house. Because no one can consistently follow a fixed schedule every day, an occupancy simulator was devised and utilized to consider the random nature of the occupancy in a typical single-family residential house. Six HVAC system control strategies were explored based on three types of thermostats (always on; schedule based; and occupancy driven) as well as two setpoint control algorithms (fixed setpoint; and adaptive control). Because the HVAC equipment in residential buildings does not usually have controllability for such fine tuning, such as multistage compressor and multi-zone air handler units, this work

assumed to use a single stage HVAC system with single-zone control. EnergyPlus was integrated into the co-simulation platform, which evaluated energy consumption and indoor temperature of a residential house in five U.S. cities (Fairbanks, New York City, San Francisco, Miami, and Phoenix) with distinctive climate zones. User's comfort level was evaluated using the adaptive model for the six different control strategies in each location. Payback period was calculated based on the local rate of utility as well as the amount of energy saving. A Co-simulation platform was explored to integrate all these aforementioned simulation entities.

4.1. Occupancy Simulator

The American Time Use Survey (ATUS)¹⁰⁶ provides detailed 24-hour diaries estimates of how, where, and with whom Americans spend their time. It completed at ten-minute resolution by over 190,000 interviews conducted from 2003 to 2017. At each 10-minute period within a day, the data includes the activity and location of the interviews so it can be used to recognize the occupancy information of a house. The blue line in Figure 27 shows the probability of occupancy in the house.

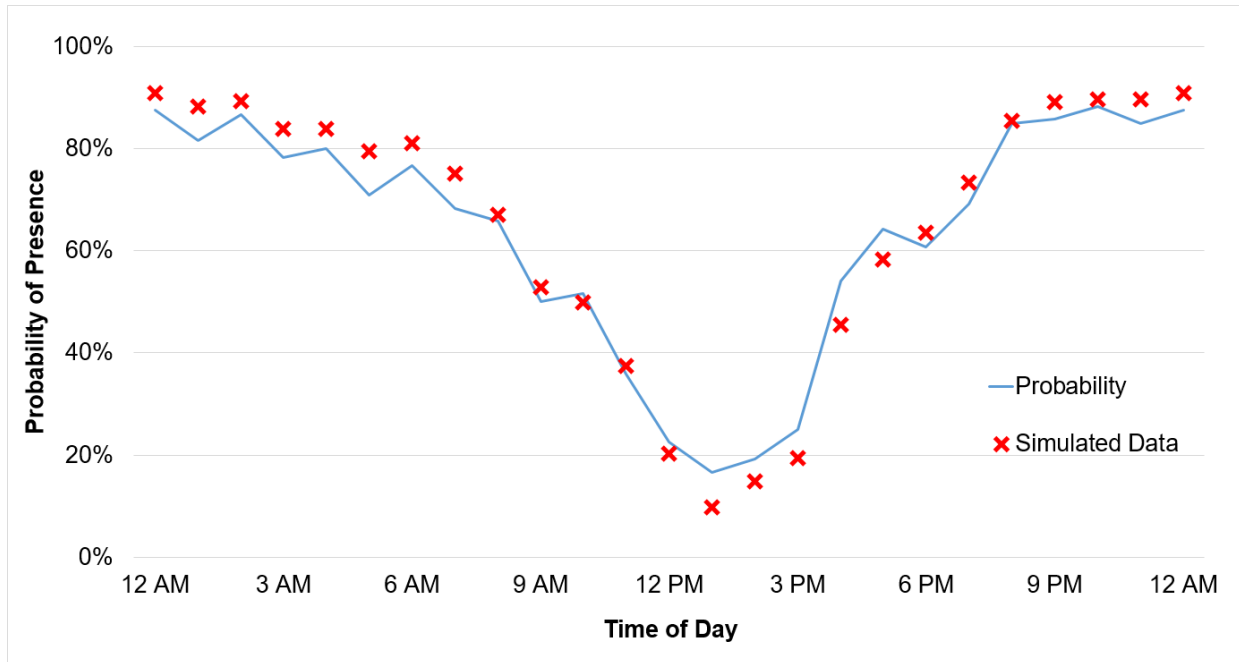


Figure 27: Demonstration of the probability of occupancy in the house at different times of day.

Since variation of occupancy pattern exists, a simple stochastic model based on the probability density function is used to model the random nature of the occupancy. The probability of occupancy in Figure 27 is compared with a random number (0 to 1) generated for each hour during the simulation. If the random number is smaller than the occupancy probability, the resident is in the house. Otherwise, no one is in the house during the time step. The flowchart of the occupancy simulator is illustrated in Figure 28. After a one-year(365 days) simulation period, the average hourly occupancy result, red cross mark in Figure 27, follows the original distribution of presence.

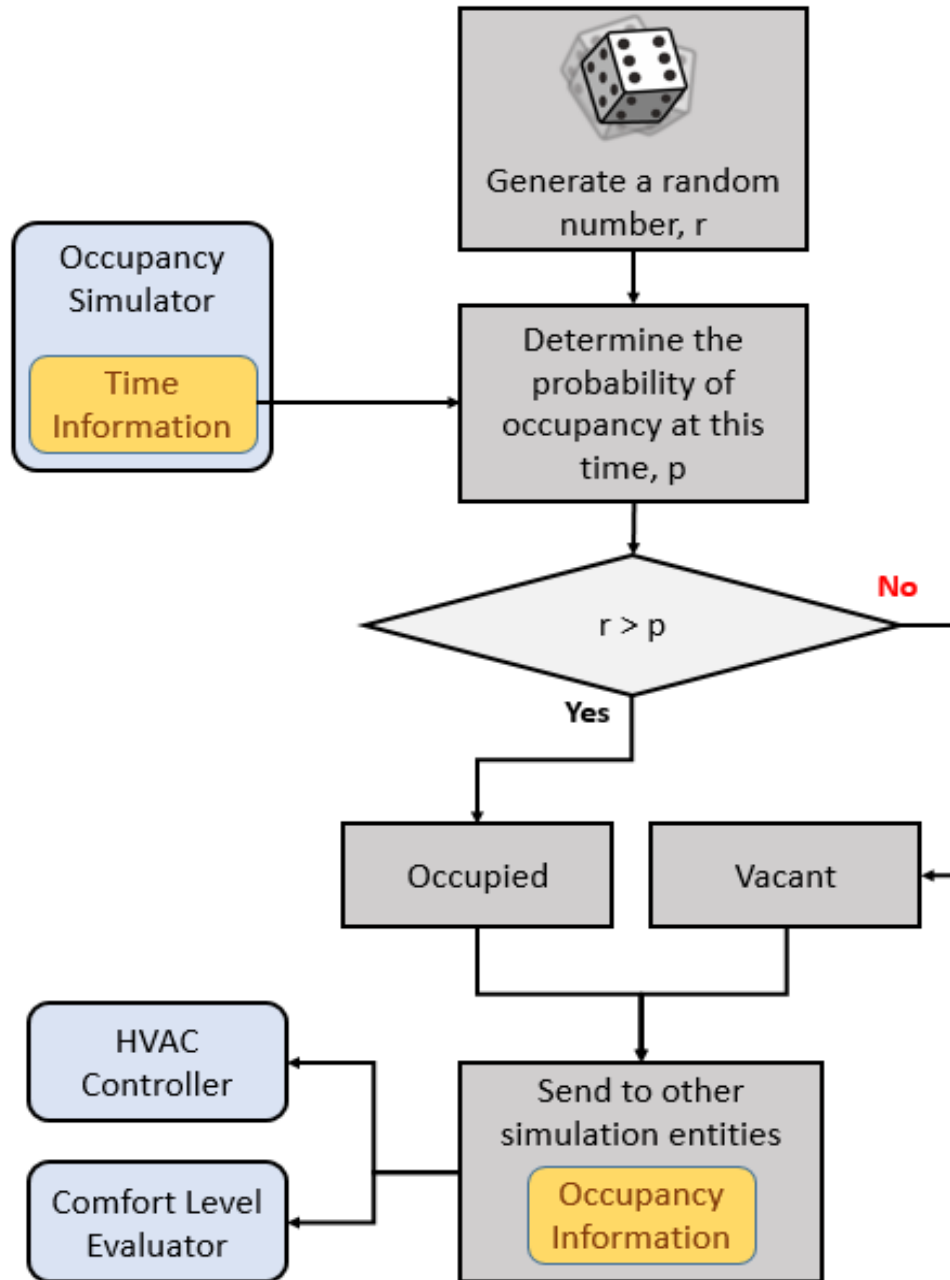


Figure 28: Illustration of the flowchart of the occupancy simulator.

4.2. Building Energy Simulation Tool

EnergyPlus,⁵² a widely adopted open source building simulation tool, can effectively model building energy consumption at each time step. The energy consumption

incorporates the physical building parameters and environment condition such as floorplan, material, HVAC system setup, and building location. The U.S. Department of Energy (DOE) provides residential prototype buildings models with different heating system type and foundation type.¹⁰⁷ The 2400 ft² two-story single house models for the 2012 edition of the International Energy Conservation Code with gas furnace heating system and crawlspace foundation type is chose as the basic building model in this work. A direct expansion cooling coil and condensing unit is used for cooling, and a natural gas furnace is utilized for heating. The footprint of the model does not change depending on location, but the materials and wall structures of the house are varied to take care of local climate conditions in the models. Five different cities (Fairbanks, New York City, San Francisco, Miami, and Phoenix) have been chosen to cover the diverse climate zones defined by RECS 2009¹⁰⁸ in the USA. To handle the different weather conditions and provide an indoor comfort environment to the occupants, the size of the HVAC system vary in different cities. The auto-size function in Energy Plus tends to oversize the HVAC system, which may not be practical due to high capital cost. To determine the HVAC sizing at each location, first I determined a size that could maintain the setpoints at least 99 % of the time. Then I checked that on extreme days the duty cycle of HVAC system is less than four times per hour. After that the sizing was rounded to the nearest commercially available size. After that, I also checked the indoor air quality (relative humidity and CO₂ concentration) to make sure the flow rate of the HVAC system is able to keep the air quality healthy and comfortable, which requires relative humidity between 25 % to 70 % and

CO₂concentration lower than 1000ppm. The detailed weather features and corresponding HVAC size in these climate zones are listed in Table 11.

Table 11: Summarization of the detailed weather features of the chosen cities from five different climate zones.

Location	Climate Zone by RECS	Note	HVAC size [k•Btu/h*]	
			Heating	Cooling
Fairbanks	Subarctic	Outdoor temperature in winter can be lower than - 30 °C	42	18
New York City	Cold	Outdoor temperature in winter is around -10 °C	18	14
San Francisco	Marine	Environment temperature change is significant during the year	5	8
Miami	Hot-Humid	Relative humidity is high need extra energy to handle it.	5	18
Phoenix	Hot-Dry	Hottest outdoor temperature can be 50 °C in summer.	8	27

*1000 Btu/h = 293 W

4.3. Thermal Comfort Standard

ASHRAE 55⁶² introduced the adaptive comfort model to estimate the occupant's comfort level. The adaptive model is based on the idea that outdoor climate influences indoor comfort because humans can adapt their clothing level to indoor and/or outdoor thermal conditions during different times of the year. Researchers survey building occupants about their thermal comfort while taking simultaneous environmental

measurements. Analyzing a global database of 21,000 measurements showed that occupants' accepted or preferred temperature range depends on outdoor conditions. These results were incorporated in the ASHRAE 55-2004 standard as the adaptive comfort model.

Figure 29 presents the detailed indoor HVAC operative temperature range based on the mean outdoor air temperature. The comfortable zone (blue grid area) is the temperature range which satisfies 90 % of the people, the acceptable zone (wider yellow dashed area) is the temperature band where 80 % of the people feel comfortable. Out of the acceptable zone is treated as the uncomfortable zone. The green solid area indicates the ideal room temperature range with the traditional setpoint, fixed between 21 °C and 23 °C.

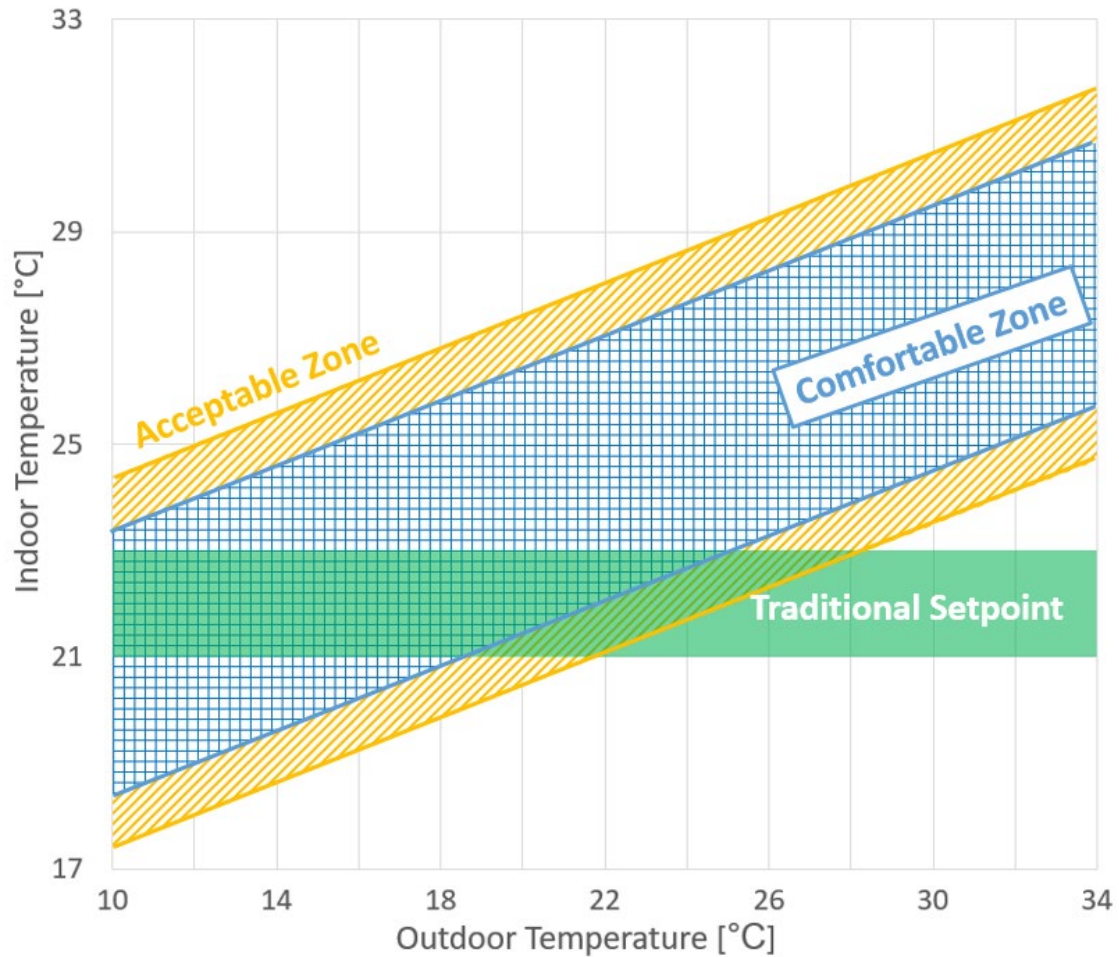


Figure 29: Acceptable operative temperature ranges based on mean outdoor temperature defined by adaptive model. The comfortable zone (blue grid area) and acceptable zone (yellow pattern area) depict the temperature ranges accepted by 90 % and 80 % of people accordingly.

Figure 30 shows the flowchart for the comfort level evaluator. After receiving the occupancy information from occupancy simulator and outdoor temperature from building simulator, the comfort level evaluator will determine the temperature range of the comfortable/ acceptable/ uncomfortable zone based on Figure 29. By comparing the indoor temperature with the three different zones, the occupancy evaluator will record the current comfort level information in the database.

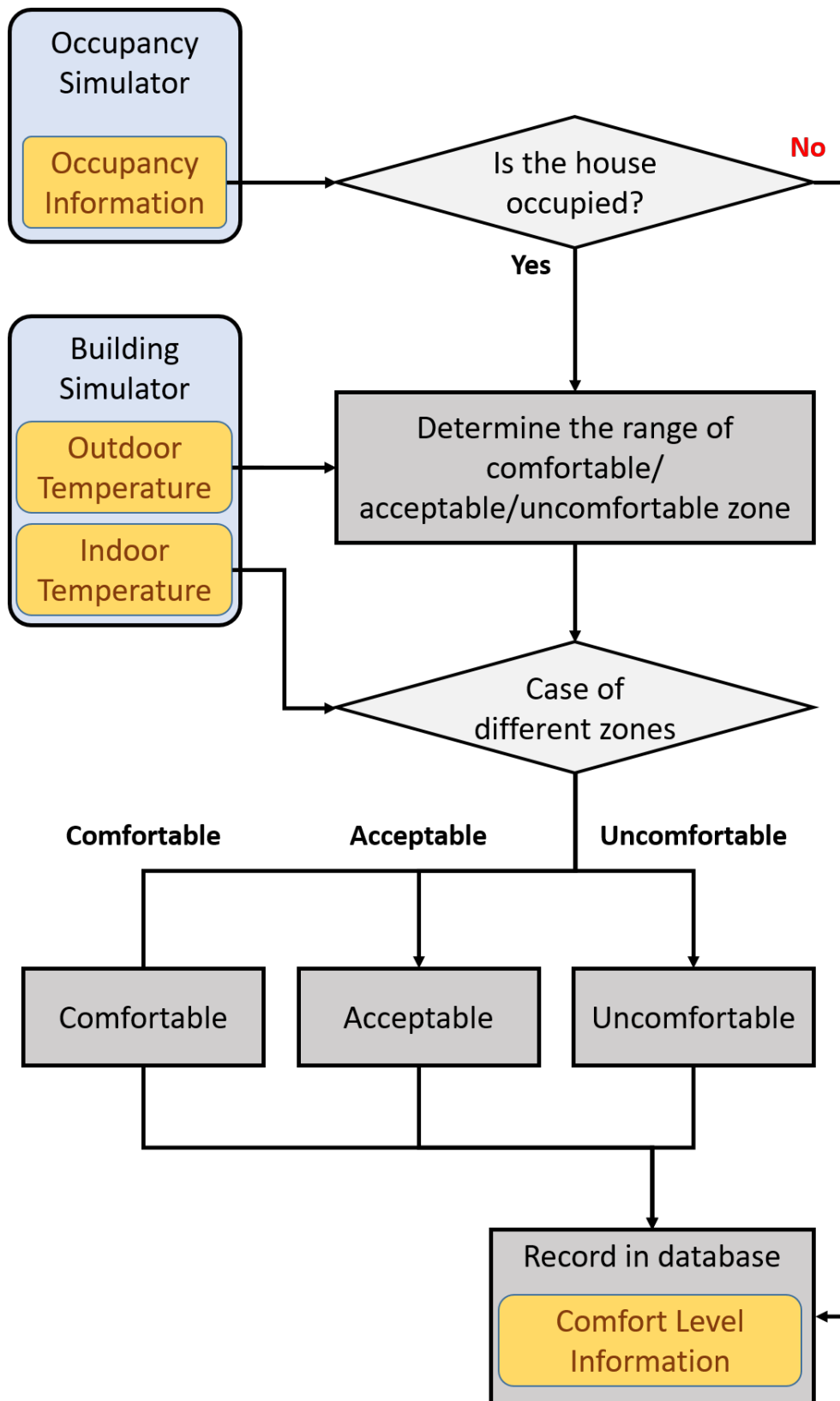


Figure 30: Display the flowchart of the comfort level evaluator.

4.4. Traditional Thermostat Control

Three different control algorithms are simulated in this paper. First is the most basic thermostat, which continuously operates to accommodate room temperature at a fixed setpoint based on a conventional comfort zone (the heating set point at 21 °C and cooling set point at 23 °C).⁶² This type of the thermostat cannot change operation schedule unless setpoint is manually changed. The second type is a schedule-based thermostat algorithm, which switches the HVAC on and off with a pre-defined schedule. The daily operation schedule is defined based on the probability of occupancy (Figure 27): The HVAC system operates for the time period with the probability greater than 0.6 (between 5 PM to 9 AM on the next day). This schedule is not optimized for a specific house or region, and the actual energy saving and user comfort can change with different schedules. The third one is the occupancy-driven algorithm, which operates the HVAC system based on the current occupancy information. It turns on the HVAC system when the house is occupied and turns it off right after the occupant leaves. All of the control algorithms have heating and cooling set points at 21 °C and 23 °C, respectively. Since some thermostats do not allow turning off, the setback temperature are set at its lowest and highest setpoints, 55 °F (12 °C) and 90 °F (32 °C). The detailed HVAC control algorithms are listed in Table 12 and Figure 31. To avoid turning the HVAC system on and off too frequently, the actual HVAC setting inputted into EnergyPlus are +/- 0.5 °C from the heating/cooling setpoint determined by the algorithm. As a result, the room temperature fluctuated around the setpoint in the range of 1 °C.

Table 12: Summary of the HVAC system setpoint based on three different control algorithm and operation conditions. P is the probability of occupancy.

Operation condition	Heating set point [°C]	Cooling set point [°C]
A. Always on		
Always	21	23
B. Schedule based		
9AM- 5PM (OFF) $P < 0.6$	12	32
5PM-9AM (ON) $P > 0.6$	21	23
C. Occupancy driven		
Current with occupant (ON)	21	23
Current without occupant (OFF)	12	32

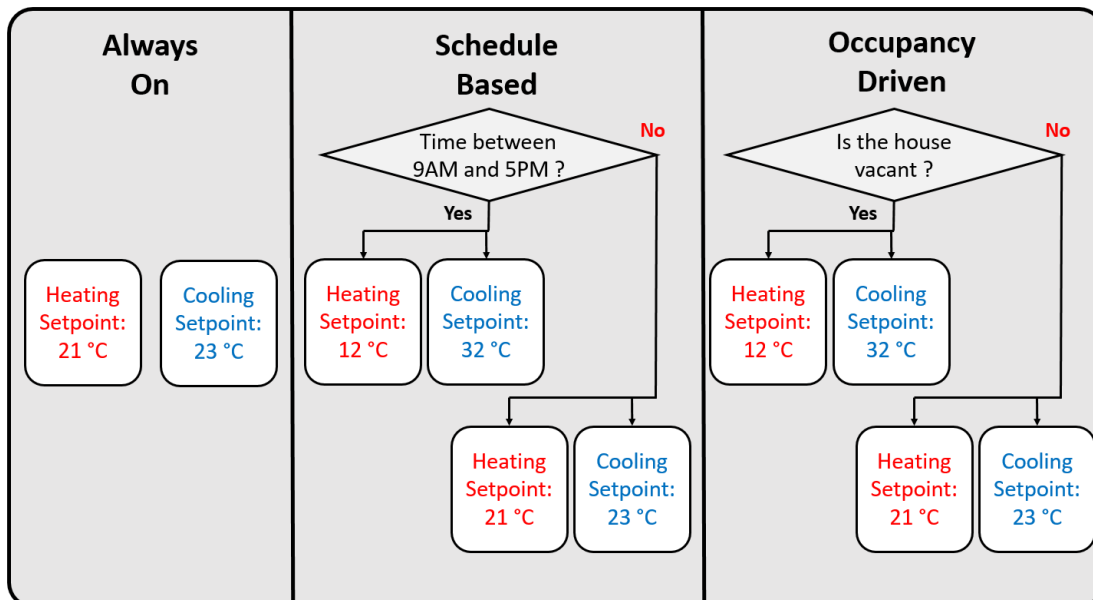


Figure 31: Demonstrate the flowchart of the HVAC system controller based on three different control algorithm and operation conditions.

Figure 32 demonstrates heating setpoint data of the occupancy-driven control algorithm on a winter day. From the plot, the heating setpoint changes based on the occupancy information and indoor room temperature fluctuates within 0.5 °C from the setpoint (21 °C). At 14:00, due to the cold weather, the HVAC system takes up to an hour to increase the room temperature from 12 °C to 21 °C, occupant feels comfort after 14:15 (room temperature greater than 18.5 °C). Because the HVAC system is turned off during non-occupied period, as shown by natural cooling of the house until the setback point (12 °C), unnecessary energy usage can be saved.

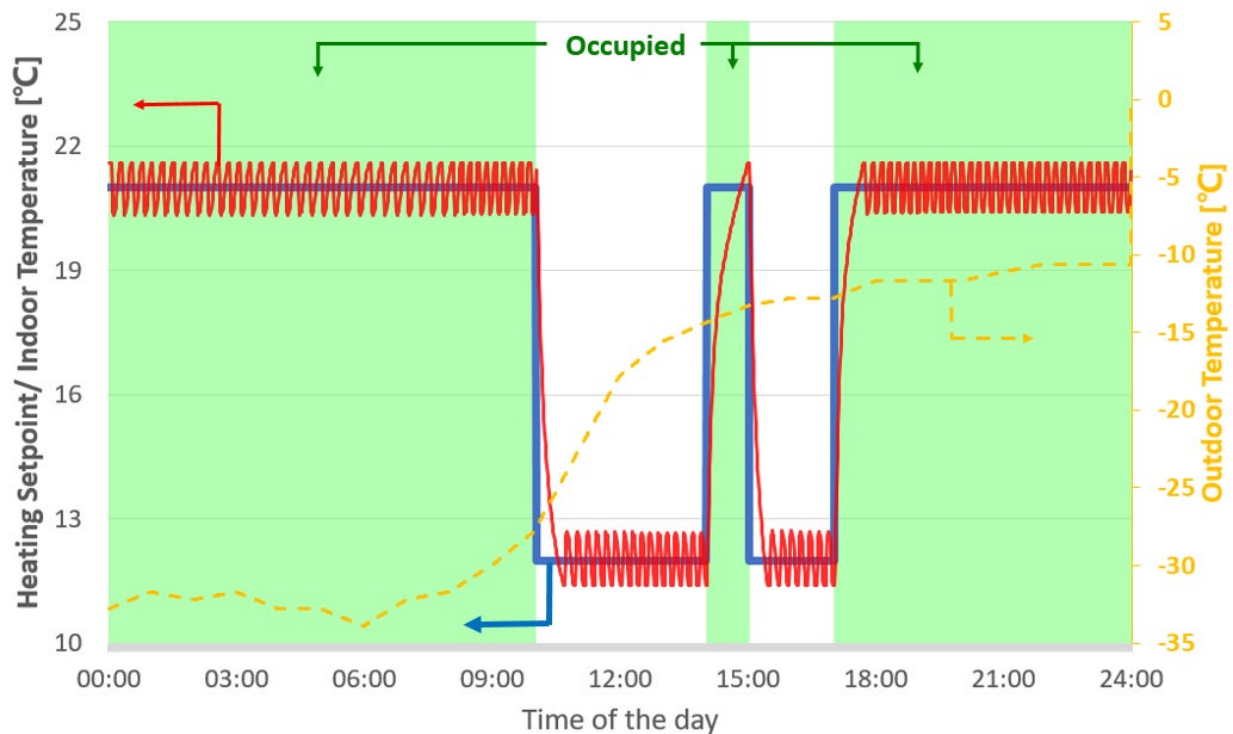


Figure 32: Illustration of the occupancy-driven control algorithm in a cold day at Fairbanks, AK. The outdoor temperature is displayed in yellow dash line. The HVAC system turns on and off based on the occupancy information (shaded area), and the room temperature (red thin solid line) fluctuates around the setpoint (blue thick solid line) when the room is occupied.

4.5. Adaptive Thermostat Control

In the simulation, the comfort level was determined based on adaptive comfort model, shown in Figure 29, although the temperature setpoint was fixed between 21 °C and 23 °C. The traditional setpoint is unable to provide a comfort environment (too cold) to the occupants with additional energy consumption when the outdoor temperature is high. In order to provide a more comfortable environment for the users without unnecessary energy consumption, an adaptive thermostat control algorithm was explored, which changes the setpoint based on the outdoor temperature. This work set the heating and cooling setpoint 0.6 °C higher and lower than the border between the comfort zone and acceptable zone, and the fluctuation of the indoor temperature was limited to 0.5 °C. As a result, the room temperature will remain inside of the comfort zone. The flowchart of the HVAC controller with adaptive setpoints is displayed in Figure 33. The saving impact and comfort level of this adaptive thermostat is also simulated and shown in the result section.

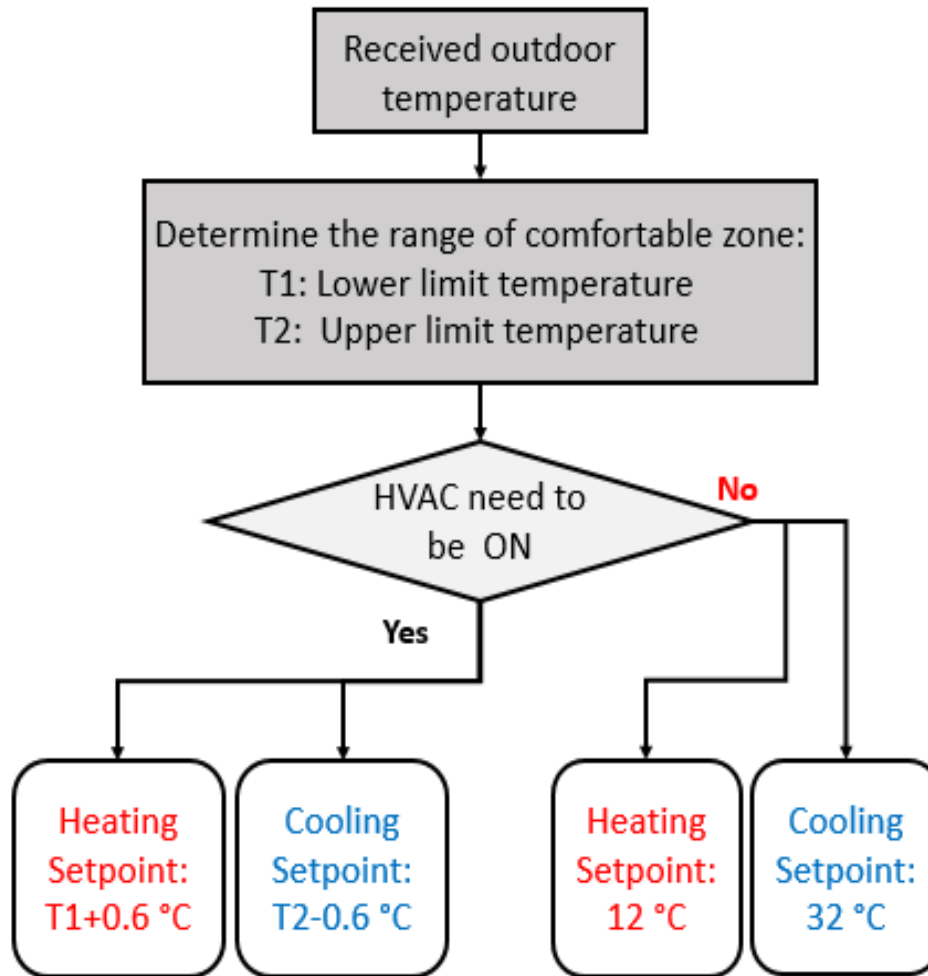


Figure 33: The flowchart of the HVAC controller with adaptive setpoints based on different operation conditions.

4.6. Co-Simulation Platform

The National Institute of Standards and Technology (NIST) developed an open-source Cyber-Physical System (CPS) experiment and testing environment called Universal CPS Environment for Federation (UCEF). Roth et al.¹⁰⁹ introduced how UCEF integrates multiple different simulation software, federates, coded in different operating system and development environments, which makes co-simulation with different software doable and straightforward. UCEF accomplish timing

synchronization and data transfer among a group of federates by utilizing the IEEE's High Level Architecture (HLA) standard.¹¹⁰ It leverages the IEEE's HLA standard for its communication protocol, implemented by the Portico.¹¹¹

EnergyPlus has a preset co-simulation interface through the Functional Mock-up Interface (FMI) standard created by Modelisar.¹¹² The standard connects EnergyPlus simulation platforms to an external model by using a Functional Mock-up Unit (FMU). In the previous work¹¹³, we interfaced the FMU with UCEF via TCP/IP. For each time step, the EnergyPlus sends the time information to the occupancy simulator, which in turn provides simulated occupancy information to the HVAC controller. The HVAC controller determines the HVAC setpoint (cooling and heating) based on the occupancy information and outside temperature when adaptive controller is employed. EnergyPlus receives and updates the new setpoint via FMU and evaluates energy consumption until the next time step. At the same time, the comfort level evaluator checks and record the current comfort level based on the indoor and environment information from EnergyPlus and occupancy information from the occupancy simulator. The detailed schematic graph is shown below in Figure 34.

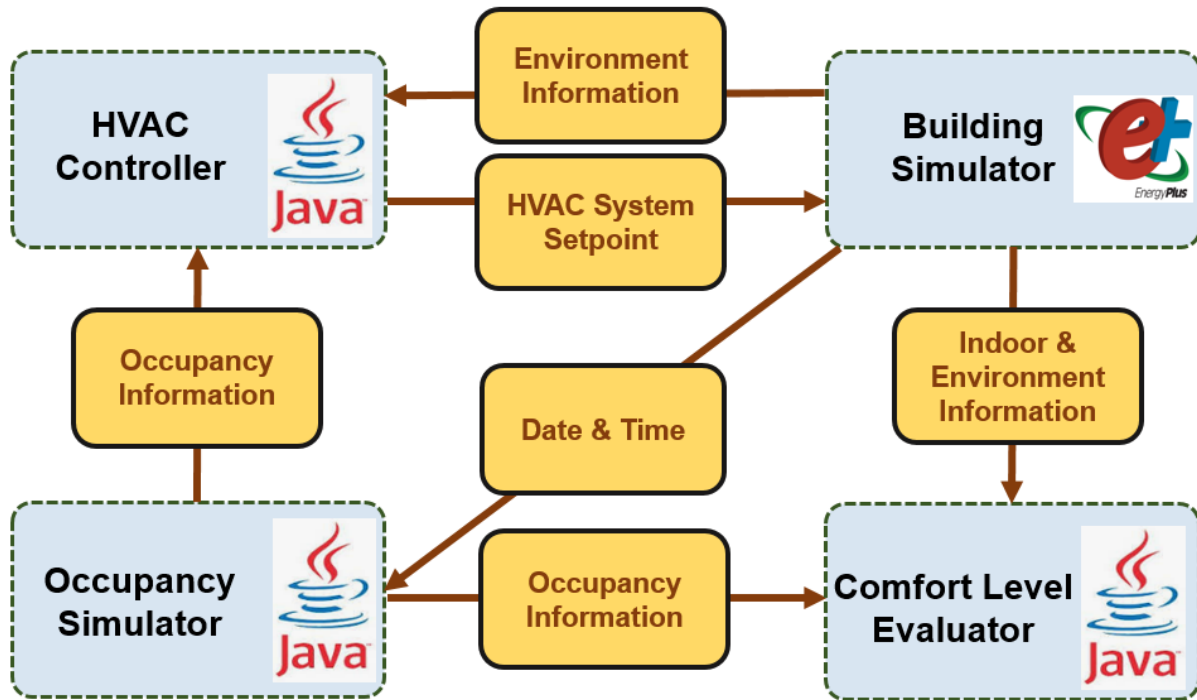


Figure 34: Representation the schematic of data transfer between building simulator, occupant simulator and HVAC controller.

4.7. Results & Discussion

Figure 35 illustrates annual simulation results of New York City for six different control strategies. The comfort level presents the ratio reflecting if the indoor temperature is comfortable, acceptable or uncomfortable when the building is occupied. With the fixed setpoint control, the always-on thermostat can only provide 93 % comfort ratio, because 21-23 °C is excluded from the comfort zone when the outdoor temperature is higher than 26 °C (Figure 29). The schedule based thermostat saved 5 MWh of energy but significantly increased the uncomfortable ratio by 12 %. An occupant may feel uncomfortable when one does not come back as schedule. Schedule based thermostat may have larger energy saving potential but risk more discomfort. As

mentioned before, this schedule is just one setting and the energy saving and comfort level may vary with other schedules. By contrast, the occupancy-driven thermostat was able to improve this situation by turning on the system simultaneously when the occupant is back, raising the comfort ratio to 90 %. Due to thermal mass of the house, an occupant may feel uncomfortable in the beginning of occupant period, which corresponds to the 3 % uncomfortable ratio. In general, the occupancy-driven thermostat can save about 25 % of the energy consumed by the always-on thermostat without increasing user's uncomfortable level.

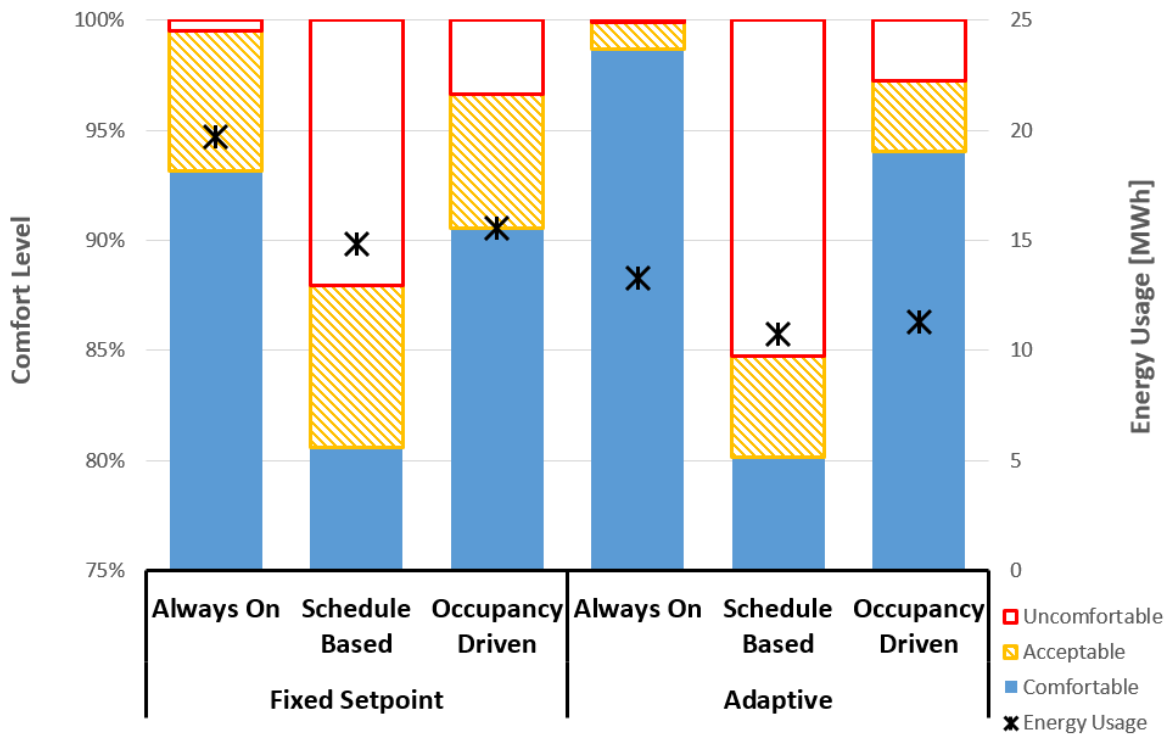


Figure 35: Demonstration of the result of annual comfort result and energy consumption at New York City, NY. The stacked bar chart shows the ratio of comfort zone (blue shaded area), acceptable zone (yellow pattern area) and uncomfortable zone (red blank area). Annual energy usages are displayed with black star marks and y-axis on the right.

With the adaptive control, the comfort ratio of the always-on thermostat is increased to 99 %. The 1 % acceptable ratio is attributed to the standard of choosing the size of HVAC system, which is undersized for few days with extreme weather condition. The total uncomfortable ratio has been reduced to almost 0 since the adaptive control algorithm sets the setpoint inside of the comfortable zone. Energy saving impact by adaptive control is larger than having an occupancy information alone. Uncomfortable ratio is decreased, and the reduction in energy consumption is quite significant. The always-on thermostat with adaptive control can save about 30 % of the energy in New York City as shown in Figure 35. Occupancy information additionally provides energy saving of 15 %.

The detailed simulation results of all five cities are tabulated in Table 13. The energy consumption of the always-on thermostat with the fixed setpoint control was used as the baseline to evaluate the energy saving impact of the control strategies. Similar trend has been observed in different locations that with the adaptive control – always on thermostat saves more energy (between 14 % and 54 %) than occupancy driven thermostat with fixed setpoint, while it achieves similar or better thermal comfort level. The occupancy-driven adaptive thermostat save 8 % more. Since 21 °C - 23 °C is uncomfortable when the outdoor temperature is high, the adaptive control could dramatically increase the comfort level in the hot climate areas (Miami and Phoenix).

Table 13: Comparison of the annually comfortable level, energy saving impact, and energy usage in Fairbanks, New York City, San Francisco, and Miami Phoenix.

			Comfortable Ratio [%]	Acceptable Ratio [%]	Uncomfortable Ratio [%]	Saving Impact	Energy Usage [MWh]
Fairbanks	Fixed Setpoint	Always On	95.5	1.7	2.8	0%	37.41
		Schedule Based	78.2	3.5	18.3	13%	32.65
		Occupancy Driven	89.5	2.2	8.4	11%	33.13
	Adaptive	Always On	98.6	0.6	0.8	14%	32.08
		Schedule Based	76.8	3.5	19.6	20%	29.90
		Occupancy Driven	88.7	2.3	9.0	20%	30.00
New York City	Fixed Setpoint	Always On	93.2	6.3	0.5	0%	19.70
		Schedule Based	80.6	7.3	12.1	25%	14.83
		Occupancy Driven	90.6	6.1	3.4	21%	15.53
	Adaptive	Always On	98.7	1.2	0.1	33%	13.29
		Schedule Based	80.2	4.6	15.2	45%	10.76
		Occupancy Driven	94.0	3.3	2.7	43%	11.31
San Francisco	Fixed Setpoint	Always On	99.6	0.3	0.1	0%	8.23
		Schedule Based	90.5	4.3	5.3	38%	5.08
		Occupancy Driven	92.0	4.2	3.7	34%	5.41
	Adaptive	Always On	98.7	1.0	0.3	54%	3.78
		Schedule Based	89.9	4.8	5.3	65%	2.90
		Occupancy Driven	94.9	3.3	1.8	64%	3.00
Miami	Fixed Setpoint	Always On	58.8	32.7	8.5	0%	29.48
		Schedule Based	61.5	21.8	16.7	35%	19.15
		Occupancy Driven	70.5	23.9	5.7	28%	21.28
	Adaptive	Always On	94.7	5.2	0.1	54%	13.51
		Schedule Based	77.3	5.9	16.8	67%	9.80
		Occupancy Driven	90.7	7.2	2.2	63%	10.78
Phoenix	Fixed Setpoint	Always On	65.6	14.9	19.5	0%	26.03
		Schedule Based	66.3	13.7	20.1	25%	19.47
		Occupancy Driven	69.2	13.9	17.0	21%	20.44
	Adaptive	Always On	89.9	10.9	0.2	43%	14.93
		Schedule Based	76.8	10.7	12.5	49%	13.26
		Occupancy Driven	86.5	11.4	2.1	49%	13.37

Since the energy saving and comfort level is not guaranteed with the schedule based thermostats (highly dependent on how the user set his schedule), the work also examine energy saving impacts (Figure 36) and economic benefits (Table 14) of three control strategies; fixed setpoint control with occupancy driven thermostat, adaptive control with always on thermostat, and adaptive control with occupancy-driven thermostat. Occupancy information can save between 11 % and 34 %, while adaptive control alone can save between 14 % and 54 %. Hence, adaptive control is a more effective way of saving energy regardless of climate zones. Such saving impact is more apparent in a warmer climate, as there is more saving potential in cooling with adaptive control (Figure 29). However, occupancy information can save additionally with adaptive control, and is more effective in colder climate. In Fairbanks, occupancy information contributes 28 % to the total saving impact of adaptive control with occupancy-driven thermostat. This ratio decreases to only 12 % in Phoenix. Although energy saving ratio in cold climate zones seems to be low, actual amount of energy saving can be larger. However, such saving does not directly correspond to economic benefits due to different utility rates or tariff.

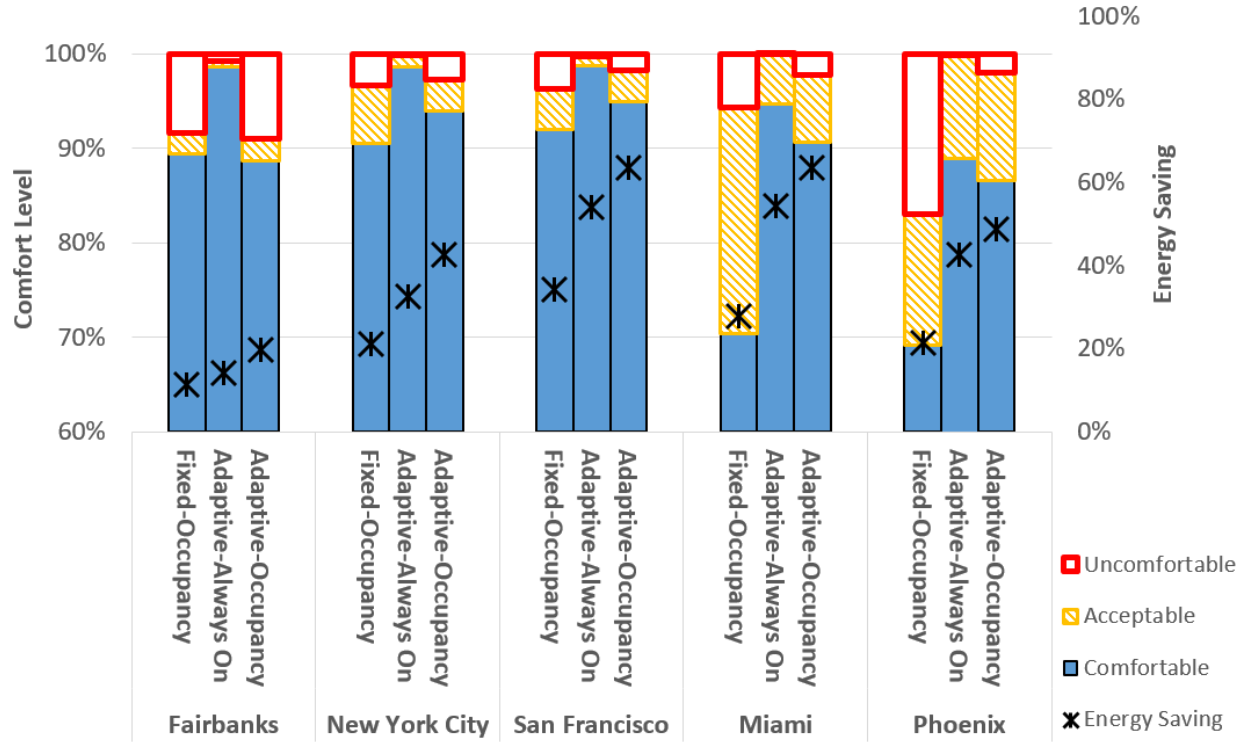


Figure 36: Comparison of the annual comfortable ratio and saving impact of the Fixed setpoint - Occupancy, Adaptive - Always On, and Adaptive - Occupancy in five different locations.

Table 14 shows the monthly saving and payback period with respect to two energy usage behaviors: 1) most wasteful behavior (fixed setpoint – always on); and 2) energy conscious behavior without a smart thermostat (fixed setpoint – schedule based). Monthly saving was evaluated using an average electricity rate/tariff for cooling load and gas rate/tariff for heating load in each city.^{114–118} Payback period¹¹⁹ was evaluated based on the capital cost of \$300 and the discount rate of 6 % using Eq. (4.1):

$$P = \frac{\ln \left(\frac{M}{(M - Cr)} \right)}{\ln (1 + r)} \quad (4.1)$$

Where P is the payback period, C is the capital cost (\$300), M is the monthly saving and r is the monthly discount rate (0.5 %/month). The prototype cost of the proposed one-entrance and one-bathroom approach is just about \$100. Preliminary cost estimation is posted in Appendix I. Because the operating costs and marketing costs are hard to evaluate now, the capital cost is set at \$300, which is the average price of a smart thermostat in the market. Compared to energy consumption with always on – fixed setpoint thermostat, the payback periods of three different control strategies are within a year in all locations but Fairbanks, due to low natural gas rate (\$0.1427 /therm). Compared to energy consumption with schedule based - fixed setpoint thermostat, occupancy information alone does not result in economic benefit but more comfortable ratio. Only with adaptive control, users with energy conscious behavior can get financial benefit in the most cities except Fairbanks. The payback period in San Francisco is almost four and half years, due to lower amount of energy consumption than other zones. More economical solution or incorporation of occupancy information will be necessary for shorter payback periods in San Francisco or Fairbanks. The payback period by the adaptive control is generally shorter in warmer climate than in colder climate zones due to more saving ratio. As outdoor temperature is easier to be integrated into a smart thermostat than an accurate occupancy information, adaptive control can lead to more tangible energy saving impact in an economically viable package.

Table 14: List of monthly saving and payback period of three different control strategies (Fixed setpoint - Occupancy, Adaptive - Always On, and Adaptive – Occupancy), compared with the most wasteful control strategy (Fixed setpoint - Always On) and energy-conscious control strategy (Fixed setpoint – Schedule Based).

Discount Rate:		6%/year		Capital Cost:	\$300	
Compare with Traditional Always On						
Location	Traditional Occupancy Driven		Adaptive Always On		Adaptive Occupancy Driven	
	Monthly Savings[\$]	Payback Period [Month]	Monthly Savings[\$]	Payback Period [Month]	Monthly Savings[\$]	Payback Period [Month]
San Francisco	34.32	9.0	48.98	6.2	58.67	5.2
New York City	50.77	6.0	79.43	3.8	99.66	3.0
Phoenix	57.42	5.3	106.37	2.8	119.31	2.5
Miami	82.10	3.7	159.55	1.9	186.92	1.6
Fairbanks	15.37	20.6	22.52	13.8	32.65	9.4
Compare with Traditional Schedule Based						
Location	Traditional Occupancy Driven		Adaptive Always On		Adaptive Occupancy Driven	
	Monthly Savings[\$]	Payback Period [Month]	Monthly Savings[\$]	Payback Period [Month]	Monthly Savings[\$]	Payback Period [Month]
San Francisco	-9.10	Never	5.55	54.0	15.24	19.7
New York City	-11.23	Never	17.43	17.2	37.67	8.0
Phoenix	-10.58	Never	38.37	7.8	51.31	5.8
Miami	-21.65	Never	55.80	5.4	83.17	3.6
Fairbanks	-7.31	Never	-0.16	Never	9.97	30.1

This chapter demonstrates energy savings and economic benefits of occupancy-driven control strategies for residential buildings by utilizing a co-simulation tool, UCEF. An occupancy simulator was developed to consider the random nature of the occupancy. A building energy software, EnergyPlus, was used to estimate the energy consumption in the test building model. Six HVAC system control strategies were

explored based on three types of thermostats (always on; schedule based; and occupancy driven) as well as two setpoint control algorithms (fixed setpoint; and adaptive control). In order to take the occupant's comfort into consideration, a comfort level evaluator was devised based on the adaptive comfort model. A one-year long simulation had been repeated in five U.S. cities (Fairbanks, New York City, San Francisco, Miami, and Phoenix) with distinctive climate zones. The results show occupancy information can lead to an average of 20 % additional saving regardless of setpoint control algorithms. Moreover, the economic analyses indicate that the payback period for an adaptive occupancy driven thermostat can be less than 10 months for the capital cost of \$300.

Chapter 5. Conclusion & Future Work

In this thesis, a preliminary approach of occupancy presence detection with a limited number of non-intrusive environment sensors was proposed and tested. There are two layers in the scheme: (1) Human activities layer: human activities detection models that detect occupants' activities (door handle touch, water usage, and presence in entrance) from the environmental data (temperature and infrared), and (2) ML based data fusion layer: occupancy presence detection model that utilizes ML algorithm (Random Forest, Decision Tree, K-Nearest Neighbor, and Support Vector Machine) to retrieve the real-time occupancy change event (entering, leaving and no change event) from a series of human activities.

Human activity detection models deployed four temperature sensors and one infrared sensor at five specific locations in the house. Activity detection methods were developed to relate temperature change rates of door handle/water pipe to common human activity (door handle touch and water usage). The detection method measured the duration when normalized temperature change rate is greater than the threshold, and reported as a human activity if the duration is more than one second. The experimental results showed that 100 % of water usage activities and 98.5 % of touch activities were detected without any falsely triggered event by using normalized temperature change rate data.

The occupancy presence detection model tested four machine learning based classification algorithms to predict occupancy information from human activities. From the result, a valid estimation of occupancy information can be predicted by using the Random Forest algorithm with high accuracy and F1-score (both >95 %). Moreover, the number of sensors can be further decreased to three without risking the validity of occupancy prediction.

Furthermore, this thesis also simulated the energy savings and economic benefits of occupancy-driven control strategies for residential buildings. To consider the random nature of the occupancy, an occupancy simulator was developed and integrated with building energy software, self-coded HVAC controller, and comfort level evaluator via an open-source co-simulation platform. The results show that the occupancy-driven thermostat leads to a similar energy-saving level with an improved comfort ratio as the schedule-based thermostat regardless of five distinct climate zones. An adaptive thermal comfort model was implemented for the HVAC system control strategy to save energy in a residential house further. The energy-saving impact of adaptive control is more effective than the occupancy information alone. The economic analyses indicate that the payback period for an adaptive control can be less than 14 months for the capital cost of \$300. The occupancy information can lead to an average of 20 % additional saving on the top of the adaptive control.

Occupant information plays a critical role in residential buildings for intelligent control of lighting and HVAC systems. The occupancy detection scheme presented in this

thesis, in combination with advanced building energy management strategies, can obtain a significant amount of energy saving in residential buildings.

5.1. Future Work: Quantification of Occupancy

The proposed scheme is capable of effectively detecting human entering or leaving events in residential buildings. Although occupancy information is closely related to entering or leaving events, it may be different after an event involving more than one occupant. For example, two people can either leave together with only one leaving event or get out of the house separately with two leaving events. So after the first leaving event, the occupancy state of the residence is unascertainable. This current set-up is only capable of providing entering or leaving events, but determining the number of occupants in the event involves additional information. Such limitations can be resolved by combining other types of sensors and implementing trustworthy analysis, which makes the approach become not binary decision making but a more stochastic information approach.

5.2. Future Work: Personal Preference of Setpoint

The number of occupancy affects the air conditioning/heating load. But people usually don't change the heating/cooling setpoint based on the number of the occupants. The study from Xin Jin et al.¹²⁰ also mentioned the heating/cooling setpoint can be affected more by personal preference if multiple people are in the same house, rather than

number of occupants itself. Door handle touch event detection has the potential to estimate the user's body temperature, which is highly related to comfort zone preference. The current set-up just simply assumes people have the same comfort zone preference in all the time. But a person with a higher skin temperature usually prefers a cooler room temperature than one with lower skin temperature.⁶³ The body temperature and touching habit of different people is different which may show different temperature change patterns.

For example, Figure 37 shows a set of measurement data with multiple touches from two users. The pink area shows the touches from User A and green area denotes the touches from User B. From the figure, I can see that the touches from user B have a larger peak value of normalized temperature change. The potential reason can be User B has a higher skin temperature than User A. The shape of the curve is also different. During the touch event from user A, the normalized temperature change is simply increased from 0 to the peak and decreased back to 0 after that. But for the user B touch event, the normalized temperature change has a sudden drop after the peak and reached the 2nd peak before going back to 0. So from the shape and the peak value identity information can be retrieved. This information leads to more precise comfort zone estimation due to everyone having their own preference of comfort zone.

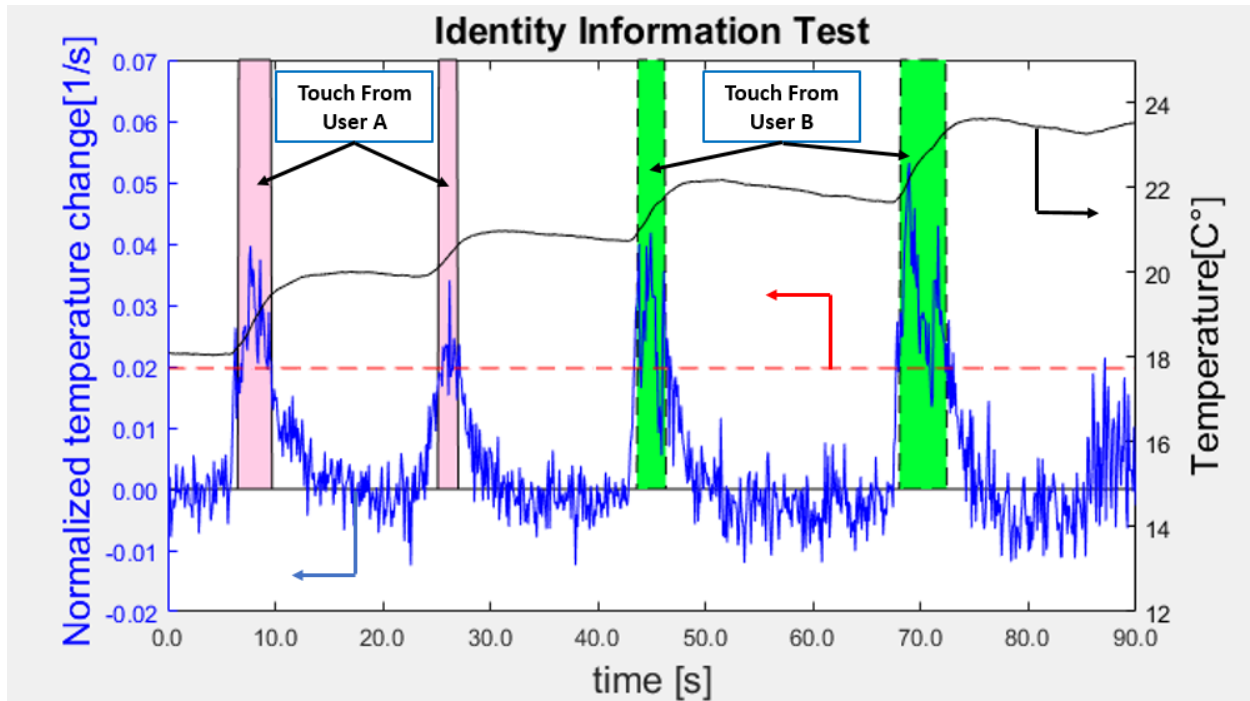


Figure 37: Demonstration of a set of temperature measurement data of a door handle with multiple touches from two users.

5.3. Future Work: Other Human Activity Detection Sensor

Door handle touch and water usage are not the only human activities during occupancy or during the changes to occupancy. Some other human behavior can also be detected from non-intrusive and easy-access information.

For example, power usage can be simply collected from a smart meter or by installing a current sensor. The total power consumption may not be related to occupancy information directly because some energy-hungry appliances (such as refrigerator,

dishwashers, washer, and dryer) still keep using energy even the building is unoccupied. However, the detection model can just focus on monitoring power usage of some specific appliance, which requires an operation signal from the people (such as television). If the model detects TV is turning on at this moment, there is a high possibility of someone in the home.

Wi-Fi data can be another easy-access data that is related to occupancy information. In order to consider the privacy concern, the model may only collect the data of the number of the collected devices and the current amount of internet traffic. There is a big chance of someone enter the building when both of the number of the collected devices and the current amount of internet traffic are increasing.

Smart camera is also a powerful sensor. With some object recognition technology, a smart camera is already able to detect and track human beings present in the video, which is highly related to entering/leaving events. However, privacy is still the first consideration. If a product can fully address the privacy concern on smart cameras, this type of sensor can play a significant role in our work. For example, a smart camera can be installed to monitor the main entrance of the building. By detecting human in the view and tracking the movement a human, the camera is able to identify it's either an entering event or a leaving event. In order to preserve users' privacy, the smart camera can process video locally and only send out a piece of non-intrusive data. In this case, just with one sensor, we are able to collect high validity data of occupancy change and the number of people in the event.

5.4. Future Work: Commercialization

The final goal of this work is to provide a commercial development of occupancy detection system in residential building. There are still some challenges associated with this goal.

The transferability of the proposed approach still needs to be further verified. Human activity detection models are normalized to handle different ambient environments. Such models have been tested at three other locations at Santa Clara University at different times of the year. The result shows that the models can still effectively detect the door handle touches and water usages. The ML model has only been tested in the Solar House. Although human behaviors are supposed to be general, e.g., people always touch the outdoor handle first to enter the house. The activity pattern may change due to personal habits, culture, or local weather, e.g., someone pull door handles to close a door; others prefer shut door with their feet. Various on-site tests across the country are required to show the proposed model's generality.

The approach is developed under the assumption that human activity sensors can detect all the activities correctly, which cannot be promised in reality. For example, due to COVID-19, more and more people start to use their elbow to pull a door handle instead of their hand. These undetected human activities will cause ML to make a poor estimation of occupancy change. By implementing trustworthy analysis, the approach can use the later happening event to correct the previous wrong decision. The scheme can calculate a trust factor for each decision and use the high trust event to review the previous decision. For example, a water usage event happens in the

house, which is supposed to be vacant. Since the water usage event has a higher trust factor, the scheme would correct the presence state immediately and decrease the previous leaving event's trust factor. Next time, when the same series of human activities happens, the scheme will have less probability to predict it as a "leaving event."

How to handle power outages or network outages is another challenge for commercialization. A potential solution is to propose a two-layer communication network. So each sensor packages connect with the data hub via Bluetooth, which is a stable point-to-point connection. The hub processes all the sensor information and sends occupancy state to occupancy-driven appliances via Wifi. Both the sensor package and the data hub are powered by batteries. Even in a power outage and network outage situation, the system can communicate and detect occupancy information. Once the network is recovered, the hub would simultaneously send current occupancy state to all occupancy-driven appliances.

For a new installation the ML algorithm requires one month for data collection. The model is trained by the ground truth, which may not be available in reality. Some detected human activities such as water usages and power usages can provide the label of the previous event since it's a strong proof of people in the house. To the greatest extent, the proposed system doesn't require human input to increase users' convenience, which could engage more people in adopting this technology.

Reference

1. U.S. Energy Information Administration (EIA) - Data. <https://www.eia.gov/totalenergy/data/browser/?tbl=T02.01#/?f=A&start=1959&end=2019&charted=3-6-9-12>, Jul 2020.
2. Land-Ocean Temperature Index (C). https://data.giss.nasa.gov/gistemp/graphs/graph_data/Global_Mean_Estimates_based_on_Land_and_Ocean_Data/graph.txt, Jul 2020.
3. Summary for Policymakers — Global Warming of 1.5 °C. <https://www.ipcc.ch/sr15/chapter/spm/> , Jul 2020.
4. World Sea-Level Rise Dataset | Data Catalog. <https://datacatalog.worldbank.org/dataset/world-sea-level-rise-dataset>, Jul 2020.
5. Calculate Your Solar Panel Payback Period | EnergySage. *Solar News* <https://news.energysage.com/understanding-your-solar-panel-payback-period/>, Jul 2020.
6. U.S. Energy Information Administration (EIA) - Renewable energy consumption Data. <https://www.eia.gov/totalenergy/data/browser/?tbl=T10.02A#/?f=A>, Jul 2020.
7. ENERGY STAR by the Numbers. https://www.energystar.gov/about/origins_mission/energy_star_numbers , Jul 2020.
8. LLNL Flow Charts. <https://flowcharts.llnl.gov/> , Jul 2020.
9. Residential Energy Consumption Survey (RECS) - Energy Information Administration. <https://www.eia.gov/consumption/residential/> , Jul 2020.

10. State Electricity Profiles - Energy Information Administration.
<https://www.eia.gov/electricity/state/>, Jul 2020.
11. Natural Gas 1998 Issues and Trends. <https://www.eia.gov/naturalgas/archive/056098.pdf>, Jul 2020.
12. Total Energy Annual Data - U.S. Energy Information Administration (EIA).
<https://www.eia.gov/totalenergy/data/annual/index.php>, Jul 2020.
13. List of U.S. states and territories by carbon dioxide emissions.
https://en.wikipedia.org/w/index.php?title=List_of_U.S._states_and_territories_by_carbon_dioxide_emissions&oldid=947939605, Jul 2020.
14. Heat and Cool. *Energy.gov* <https://www.energy.gov/energysaver/heat-and-cool>, Jul 2020.
15. Nguyen, T. A. & Aiello, M. Energy intelligent buildings based on user activity: A survey. *Energy Build.* 56, 244–257 (2013).
16. WBCSD, Transforming the Market: Energy Efficiency in Buildings, Survey report, The World Business Council for Sustainable Development,
<http://docs.wbcsd.org/2009/08/EEB-TransformingTheMarket.pdf>, Jul 2020.
17. Liang, Y. C., Lu, X., Li, W. D. & Wang, S. Cyber Physical System and Big Data enabled energy efficient machining optimisation. *J. Clean. Prod.* 187, 46–62 (2018).
18. Uhlemann, T. H.-J., Lehmann, C. & Steinhilper, R. The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0. *Procedia CIRP* 61, 335–340 (2017).

19. Fatimah, Y. A., Widiyanto, A. & Hanafi, M. Cyber-physical System Enabled in Sustainable Waste Management 4.0: A Smart Waste Collection System for Indonesian Semi-Urban Cities. *Procedia Manuf.* 43, 535–542 (2020).
20. Gunduz, H. & Jayaweera, D. Reliability assessment of a power system with cyber-physical interactive operation of photovoltaic systems. *Int. J. Electr. Power Energy Syst.* 101, 371–384 (2018).
21. Behl, M., Jain, A. & Mangharam, R. Data-Driven Modeling, Control and Tools for Cyber-Physical Energy Systems. in *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)* 1–10 (2016). doi:10.1109/ICCPS.2016.7479093.
22. Stamatescu, G., Stamatescu, I., Arghira, N., Calofir, V. & Fagarasan, I. Building Cyber-Physical Energy Systems. *ArXiv160506903 Cs* (2016).
23. Belafi, Z., Hong, T. & Reith, A. Smart building management vs. intuitive human control—Lessons learnt from an office building in Hungary. *Build. Simul.* 10, 811–828 (2017).
24. Liang, X., Hong, T. & Shen, G. Q. Improving the accuracy of energy baseline models for commercial buildings with occupancy data. *Appl. Energy* 179, 247–260 (2016).
25. Labeodan, T., Zeiler, W., Boxem, G. & Zhao, Y. Occupancy measurement in commercial office buildings for demand-driven control applications—A survey and detection system evaluation. *Energy Build.* 93, 303–314 (2015).
26. Lu, J. *et al.* The smart thermostat: using occupancy sensors to save energy in homes. in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10* 211 (ACM Press, 2010). doi:10.1145/1869983.1870005.

27. Li, N., Calis, G. & Becerik-Gerber, B. Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations. *Autom. Constr.* 24, 89–99 (2012).
28. Emmerich, S. J. & Persily, A. K. State-of-the-Art Review of CO₂ Demand Controlled Ventilation Technology and Application. (2001).
29. Nassif, N. A robust CO₂-based demand-controlled ventilation control strategy for multi-zone HVAC systems. *Energy Build.* Complete, 72–81 (2012).
30. Mumma, Stanley A. "Transient occupancy ventilation by monitoring CO₂." ASHRAE IAQ Applications 5.1 (2004): 21-23.
31. Ansanay-Alex, Guillaume. "Estimating occupancy using indoor carbon dioxide concentrations only in an office building: a method and qualitative assessment." 11th REHVA World Congr. Energy Eff., Smart Healthy Build (2013).
32. Stancil, B. A., Zhang, C. & Chen, T. Active Multicamera Networks: From Rendering to Surveillance. *IEEE J. Sel. Top. Signal Process.* 2, 597–605 (2008).
33. Trivedi, M., Huang Kohsia & Mikic, I. Intelligent environments and active camera networks. in *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (cat. no.0 vol. 2 804–809 vol.2* (2000).
34. Erickson, V. L. & Cerpa, A. E. Occupancy Based Demand Response HVAC Control Strategy. in *Proceedings of the 2Nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building* 7–12 (ACM, 2010). doi:10.1145/1878431.1878434.
35. Erickson, V. L., Achleitner, S. & Cerpa, A. E. POEM: Power-efficient Occupancy-based Energy Management System. in *Proceedings of the 12th International*

- Conference on Information Processing in Sensor Networks* 203–216 (ACM, 2013).
doi:10.1145/2461381.2461407.
36. DigiKey Electronics - Electronic Components Distributor. <https://www.digikey.com/>.
37. Jiang, C., Masood, M. K., Soh, Y. C. & Li, H. Indoor occupancy estimation from carbon dioxide concentration. *Energy Build.* 131, 132–141 (2016).
38. Page, J., Robinson, D., Morel, N. & Scartezzini, J.-L. A generalised stochastic model for the simulation of occupant presence. *Energy Build.* 40, 83–98 (2008).
39. Candanedo, L. M. & Feldheim, V. Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models. *Energy Build.* 112, 28–39 (2016).
40. Yang, Z., Li, N., Becerik-Gerber, B. & Orosz, M. A systematic approach to occupancy modeling in ambient sensor-rich buildings: *SIMULATION* (2013)
doi:10.1177/0037549713489918.
41. Abade, B., Perez Abreu, D. & Curado, M. A Non-Intrusive Approach for Indoor Occupancy Detection in Smart Environments. *Sensors* 18, 3953 (2018).
42. Vafeiadis, T. *et al.* Machine Learning Based Occupancy Detection via the Use of Smart Meters. in *2017 International Symposium on Computer Science and Intelligent Controls (ISCSIC)* 6–12 (2017). doi:10.1109/ISCSIC.2017.15.
43. Ortega, J. L. G., Han, L., Whittacker, N. & Bowring, N. A machine-learning based approach to model user occupancy and activity patterns for energy saving in buildings. in *2015 Science and Information Conference (SAI)* 474–482 (2015).
doi:10.1109/SAI.2015.7237185.

44. Tutuncu, K., Cataltas, O. & Koklu, M. OCCUPANCY DETECTION THROUGH LIGHT, TEMPERATURE, HUMIDITY AND CO2 SENSORS USING ANN. *Int. J. Ind. Electron. Electr. Eng.* 5, 63–67 (2017).
45. Wang, W., Lin, Z. & Chen, J. Promoting Energy Efficiency of HVAC Operation in Large Office Spaces with a Wi-Fi Probe enabled Markov Time Window Occupancy Detection Approach. *Energy Procedia* 143, 204–209 (2017).
46. Hailemariam, E., Goldstein, R., Attar, R. & Khan, A. Real-time occupancy detection using decision trees with multiple sensor types. in *Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design* 141–148 (Society for Computer Simulation International, 2011).
47. Wang, W., Chen, J. & Hong, T. Occupancy prediction through machine learning and data fusion of environmental sensing and Wi-Fi sensing in buildings. *Autom. Constr.* 94, 233–243 (2018).
48. Baker, N. *et al.* *Workshop Report on Basic Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence*. <https://www.osti.gov/biblio/1478744-workshop-report-basic-research-needs-scientific-machine-learning-core-technologies-artificial-intelligence> (2019) doi:10.2172/1478744.
49. Wagstaff, K., Cardie, C., Rogers, S. & Schrödl, S. Constrained K-means Clustering with Background Knowledge. in *Proceedings of the Eighteenth International Conference on Machine Learning* 577–584 (Morgan Kaufmann Publishers Inc., 2001).
50. Webber, C. J. S. Self-Organization of Symmetry Networks: Transformation Invariance from the Spontaneous Symmetry-Breaking Mechanism. *Neural Comput.* 12, 565–596 (2000).

51. Nest. Real Savings. *Nest* <https://www.nest.com/thermostats/real-savings/>, July 2020.
52. EnergyPlus | EnergyPlus. <https://energyplus.net/>, July 2020.
53. Hensen, Jan LM. "A comparison of coupled and de-coupled solutions for temperature and air flow in a building." *ASHRAE transactions* 105.2 (1999): 962-969.
54. Zhai, Z. J. & Chen, Q. Y. Performance of coupled building energy and CFD simulations. *Energy Build.* 37, 333–344 (2005).
55. Trcka, Marija, Michael Wetter, and Jan Hensen. "Comparison of co-simulation approaches for building and HVAC/R system simulation." Proceedings of the International IBPSA Conference, Beijing, China. 2007.
56. Wetter, M. Co-simulation of building energy and control systems with the Building Controls Virtual Test Bed. *J. Build. Perform. Simul.* 4, 185–203 (2011).
57. Trčka, M., Hensen, J. L. M. & Wetter, M. Co-simulation for performance prediction of integrated building and HVAC systems – An analysis of solution characteristics using a two-body system. *Simul. Model. Pract. Theory* 18, 957–970 (2010).
58. Liu, Y., Pan, Y. & Huang, Z. SIMULATION-BASED RECEDING-HORIZON SUPERVISORY CONTROL OF HVAC SYSTEM. in (2013).
59. Mendell, M. J. *et al.* Improving the health of workers in indoor environments: priority research needs for a national occupational research agenda. *Am. J. Public Health* 92, 1430–1440 (2002).
60. Tanabe, S., Nishihara, N. & Haneda, M. Indoor Temperature, Productivity, and Fatigue in Office Tasks. *HVACR Res.* 13, 623–633 (2007).
61. Seppänen, O. & Fisk, W. Control of temperature for health and productivity in offices. *ASHRAE Trans.* 111, 680–686 (2005).

62. Standard 55 – Thermal Environmental Conditions for Human Occupancy.
<https://www.ashrae.org/technical-resources/bookstore/standard-55-thermal-environmental-conditions-for-human-occupancy>, July 2020.
63. Fanger, P. O. Thermal comfort. Analysis and applications in environmental engineering. *Therm. Comf. Anal. Appl. Environ. Eng.* (1970).
64. de Dear, R. J. & Brager, G. S. Thermal comfort in naturally ventilated buildings: revisions to ASHRAE Standard 55. *Energy Build.* 34, 549–561 (2002).
65. Howell, W. C. & Kennedy, P. A. Field Validation of the Fanger Thermal Comfort Model. *Hum. Factors* 21, 229–239 (1979).
66. Croome, D., Gan, G. & Awbi. Thermal comfort and air quality in offices. AIVC
<https://www.aivc.org/resource/thermal-comfort-and-air-quality-offices> (2013).
67. Humphreys, M. A. & Fergus Nicol, J. The validity of ISO-PMV for predicting comfort votes in every-day thermal environments. *Energy Build.* 34, 667–684 (2002).
68. Wu, T., Cao, B. & Zhu, Y. A field study on thermal comfort and air-conditioning energy use in an office building in Guangzhou. (2018) doi:10.1016/j.enbuild.2018.03.030.
69. Parkinson, T., de Dear, R. & Brager, G. Nudging the adaptive thermal comfort model. *Energy Build.* 206, 109559 (2020).
70. Song, Y., Wu, S. & Yan, Y. Y. Control strategies for indoor environment quality and energy efficiency—a review. *Int. J. Low-Carbon Technol.* 10, 305–312 (2015).
71. Lute, P. J., and V. A. H. Paassen. "Predictive control of indoor temperatures in office buildings energy consumption and comfort." *Proc. CLIMA2000* 2 (2000): 290-295.
72. Nesler, C. Adaptive control of thermal processes in buildings. *IEEE Control Syst. Mag.* 6, 9–13 (1986).

73. Tigrek, T. Nonlinear adaptive optimal control of HVAC systems. *Theses Diss.* (2001)
doi:10.17077/etd.5elav8dm.
74. Standards 62.1 & 62.2. <https://www.ashrae.org/technical-resources/bookstore/standards-62-1-62-2>, July 2020.
75. People Counting Sensors & Door Counter Devices | Traf-Sys.
<https://www.trafsys.com/hardware/>, July 2020
76. People Counter | People Counting System | SenSource Inc. *SenSource People Counting Solutions* <https://www.sensourceinc.com/hardware/people-counters/>, July 2020
77. The Most Advanced People Counter. *Density* <https://www.density.io/hardware/>, July 2020
78. Debes, C. *et al.* Monitoring Activities of Daily Living in Smart Homes: Understanding human behavior. *IEEE Signal Process. Mag.* 33, 81–94 (2016).
79. Srinivasan, V., Stankovic, J. & Whitehouse, K. Protecting your daily in-home activity information from a wireless snooping attack. in *Proceedings of the 10th international conference on Ubiquitous computing* 202–211 (Association for Computing Machinery, 2008). doi:10.1145/1409635.1409663.
80. Smith, M. Peeping into 73,000 unsecured security cameras via default passwords. *CSO Online* <https://www.csoonline.com/article/2844283/peeping-into-73-000-unsecured-security-cameras-thanks-to-default-passwords.html> (2014).
81. Giancoli, Douglas C. *Physics: principles with applications*. Boston: Pearson, 2016.
82. Holmes, Kenneth R. "Thermal properties." *cortex (dog)* 491 (2009): 16.

83. US Groundwater Temperature - Bradley Corporation. <https://www.bradleycorp.com/sizing-tankless-water-heaters/united-states-groundwater-temperatures>, July 2020.
84. LabView: Advanced Programming Techniques, Second Edition. *CRC Press* <https://www.routledge.com/LabView-Advanced-Programming-Techniques-Second-Edition/author/p/book/9780849333255>, July 2020.
85. BeagleBoard.org - black. <https://beagleboard.org/black>, July 2020.
86. IEEE Standard for Modeling and Simulation (M & S) High Level Architecture (HLA)– Framework and Rules. *IEEE Std 1516-2010 Revis. IEEE Std 1516-2000* 1–38 (2010) doi:10.1109/IEEESTD.2010.5553440.
87. IEEE Standard for Modeling and Simulation (M S) High Level Architecture (HLA)– Federate Interface Specification. *IEEE Std 15161-2010 Revis. IEEE Std 15161-2000* 1–378 (2010) doi:10.1109/IEEESTD.2010.5557728.
88. IEEE Standard for Modeling and Simulation (M & S) High Level Architecture (HLA)– Object Model Template (OMT) Specification. *IEEE Std 15162-2010 Revis. IEEE Std 15162-2000* 1–110 (2010) doi:10.1109/IEEESTD.2010.5557731.
89. Dahmann, J. S., Fujimoto, R. M. & Weatherly, R. M. The Department of Defense High Level Architecture. in *Proceedings of the 29th conference on Winter simulation* 142–149 (IEEE Computer Society, 1997). doi:10.1145/268437.268465.
90. Lim, S. Y. & Klm, T. G. Hybrid Modeling and Simulation Methodology based on DEVS Formalism. in 188–193 (2001).
91. Zeigler, B. P., Song, H. S., Kim, T. G. & Praehofer, H. DEVS framework for modelling, simulation, analysis, and design of hybrid systems. in *Hybrid Systems II* (eds.

- Antsaklis, P., Kohn, W., Nerode, A. & Sastry, S.) 529–551 (Springer, 1995).
doi:10.1007/3-540-60472-3_27.
92. MathWorks - Makers of MATLAB and Simulink. <https://www.mathworks.com/>, July 2020.
93. Albagli, A. N., Falcão, D. M. & de Rezende, J. F. Smart grid framework co-simulation using HLA architecture. *Electr. Power Syst. Res.* 130, 22–33 (2016).
94. Varga, A. & Hornig, R. An overview of the OMNeT++ simulation environment. in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops* 1–10 (ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008).
95. Bellifemine, F., Poggi, A. & Rimassa, G. JADE- A FIPA compliant agent framework. in 97–108 (1999).
96. Forouzan, B. A. *TCP/IP Protocol Suite*. (McGraw-Hill, Inc., 2002).
97. Boser, B. E., Guyon, I. M. & Vapnik, V. N. A training algorithm for optimal margin classifiers. in *Proceedings of the fifth annual workshop on Computational learning theory* 144–152 (Association for Computing Machinery, 1992).
doi:10.1145/130385.130401.
98. Peng, Y., Rysanek, A., Nagy, Z. & Schlüter, A. Using machine learning techniques for occupancy-prediction-based cooling control in office buildings. *Appl. Energy* 211, 1343–1358 (2018).
99. Sheng-Fuu Lin, Jaw-Yeh Chen & Hung-Xin Chao. Estimation of number of people in crowded scenes using perspective transformation. *IEEE Trans. Syst. Man Cybern. - Part Syst. Hum.* 31, 645–654 (2001).

100. Powers, D. M. W. Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation. *undefined*
<https://www.semanticscholar.org/paper/Evaluation%3A-from-Precision%2C-Recall-and-F-measure-to-Powers/6d03a38c5ddb7c7cd1ceb59b28907dc918c5d83a> (2011).
101. Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A. F. & Nielsen, H. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics* 16, 412–424 (2000).
102. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12, 2825–2830 (2011).
103. Nesa, N. & Banerjee, I. IoT-Based Sensor Data Fusion for Occupancy Sensing Using Dempster–Shafer Evidence Theory for Smart Buildings. *IEEE Internet Things J.* 4, 1563–1570 (2017).
104. Singh, A. P. *et al.* Machine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes. in *2018 IEEE Globecom Workshops (GC Wkshps)* 1–6 (2018). doi:10.1109/GLOCOMW.2018.8644432.
105. de Souza Dutra, M. D., Anjos, M. F. & Le Digabel, S. A general framework for customized transition to smart homes. *Energy* 189, 116138 (2019).
106. American Time Use Survey Home Page. <https://www.bls.gov/tus/>, July 2020.
107. Residential Prototype Building Models | Building Energy Codes Program. https://www.energycodes.gov/development/residential/iecc_models, July 2020
108. Baechler, Michael C., *et al.* Building America best practices series-high-performance home technologies: guide to determining climate regions by county. EERE Publication and Product Library, Washington, DC (United States), 2013.

- 109.Roth, T. *et al.* Cyber-Physical System Development Environment for Energy Applications. in (American Society of Mechanical Engineers Digital Collection, 2017). doi:10.1115/ES2017-3589.
- 110.1516-2010 - IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-- Framework and Rules. <https://standards.ieee.org/standard/1516-2010.html>.
- 111.The Portico Project. <http://timpokorny.github.io/public/index.html>, July 2020
- 112.Functional Mock-up Interface. <https://fmi-standard.org/>, July 2020
- 113.Singer, J., Roth, T., Wang, C., Nguyen, C. & Lee, H. EnergyPlus Integration Into Cosimulation Environment to Improve Home Energy Saving Through Cyber-Physical Systems Development. *J. Energy Resour. Technol.* 141, (2019).
- 114.ENSTAR Natural Gas | All our energy goes into our customers. <https://www.enstarnaturalgas.com/>, July 2020
- 115.Florida Public Utilities -Natural Gas, Electricity, Propane. *Florida Public Utilities* <https://fpuc.com/>, July 2020
- 116.Home : Geographic Information : U.S. Bureau of Labor Statistics. <https://www.bls.gov/regions/home.htm>, July 2020
- 117.Rates / Tariff - GVEA - Golden Valley Electric Association. <http://www.gvea.com/rates/rates>, July 2020
- 118.Southwest Gas. *Southwest Gas Corporation* www.swgas.com/en/home, July 2020
- 119.Bhandari, S. B. Discounted Payback Period — A Viable Complement to Net Present Value for Projects with Conventional Cash Flows. *J. Cost Anal.* 7, 43–53 (1989).

120. Jin, X., Baker, K., Christensen, D. & Isley, S. Foresee: A user-centric home energy management system for energy efficiency and demand response. *Appl. Energy* 205, 1583–1595 (2017).

Appendices

Appendix A. Human Activity Detection Code (Matlab)

```
clear all
close all
T_f = 34; %Finger Temperature
for p=1:5
X=['TouchTest',num2str(p),'.xlsx'];
[NUM,TXT,Data] = xlsread(X);
S=size(Data);
N=S(1,1)-1;
Temp=zeros(N,4);
DT = zeros(N,4);
DTN= zeros(N,4);
Check = zeros(N,4);
Touch = zeros(N,4);
Raw=zeros(N,1);

for i=1:N
    Raw(i,1)=Data{i+1,2};
end
Time = (0.1:0.1:N*0.1)';

M=[0,2,5,10]; %moving intervals
for j =1:4
m=M(j);
for i=1:N
    if i<=m
        Temp(i,j)=mean(Raw(1:i,1));
    elseif i>N-m
        Temp(i,j)=mean(Raw(i:N,1));
    else
        Temp(i,j)=mean(Raw(i-m:i+m,1));
    end

    if i>1
        DT (i,j)=Temp(i,j)-Temp(i-1,j);
```

```

    end
    Data_T(:,p)=Temp(:,4);
    Data_DT(:,p)=DT(:,4);
end

end
MaxDT(p) = max(DT(:,4));

YL=['Temperature[C', char(0176), ''];
figure (1)
hold on
subplot (2,1,1) ,plot (Time, Temp(:,1))
xlabel ('time[s]','FontSize',16)
ylabel (YL,'FontSize',16)
ylim([18 32])
xlim([0 60])
title ('Temperature vs Time w/o Moving average','FontSize',16)
% legend('1st touch','2nd touch','3rd touch','4th touch','5th touch')

figure (1)
hold on
subplot (2,1,2) ,plot (Time, Temp(:,4))
xlabel ('time[s]','FontSize',16)
ylabel (YL,'FontSize',16)
ylim([18 32])
xlim([0 60])
title ('Temperature vs Time w/ Moving average M=10','FontSize',16)
% legend('1st touch','2nd touch','3rd touch','4th touch','5th touch')

YL=['Temperature change [C', char(0176), ''];
figure (2)
hold on
plot (Time, DT(:,4))
xlabel ('time[s]','FontSize',16)
ylabel (YL,'FontSize',16)
ylim([-0.5 0.5])
xlim([0 60])
title ('Temperature change in 0.1s vs Time w/ Moving average, M=10','FontSize',16)

end

%%
figure (1)
hold on
subplot (2,1,1)
legend('1st touch','2nd touch','3rd touch','4th touch','5th touch')

figure (1)

```

```

hold on
subplot (2,1,2)
legend('1st touch','2nd touch','3rd touch','4th touch','5th touch')

figure (2)
hold on
legend('1st touch','2nd touch','3rd touch','4th touch','5th touch')

%% Setup threshold

disp(['Maximum temperature change in 0.1s for previous touch testes'])
disp([num2str(MaxDT)])
Mean_MaxDT = mean(MaxDT);
disp(['Mean of maximum temperature change in 0.1s for previous touch testes',num2str(Mean_MaxDT)])
Std_MaxDT = std(MaxDT);
disp(['Standard deviation of maximum temperature change in 0.1s for previous touch testes',num2str(Std_MaxDT)])
disp(['For 99% accuracy of normal distribution, We set Threshold = Mean - 2.57*Std' ])
Th = Mean_MaxDT - 1.96 * Std_MaxDT;
disp(['Threshold = ', num2str(Th)])

%% Check with first 5 touch test
for i=1:5
    for j=1:N-10
        if Data_DT(j,i)>=Th
            for k=10:N-j
                if mean(Data_DT(j:j+k,i))<=Th

                    break
                else
                    Touch(j:j+k,i)=1;
                end
            end
        end
    end
end

end

%%
y1=Th*ones(1,N);
YL=['Temperature change in 0.1s [C', char(0176), ']'];

```

```

figure (3)
subplot (3,2,1), plot (Time, y1,'r')
hold on
subplot (3,2,1), plot (Time, Data_DT(:,1))
subplot (3,2,1), plot (Time, Touch(:,1),'k')
xlabel ('time[s]','FontSize',14)
ylabel (YL,'FontSize',14)
ylim([-0.5 0.5])
xlim([0 60])
title ('1st Test w/ Threshold','FontSize',14)

subplot (3,2,2), plot (Time, y1,'r')
hold on
subplot (3,2,2), plot (Time, Data_DT(:,2))
subplot (3,2,2), plot (Time, Touch(:,2),'k')
xlabel ('time[s]','FontSize',14)
ylabel (YL,'FontSize',14)
ylim([-0.5 0.5])
xlim([0 60])
title ('2nd Test w/ Threshold','FontSize',14)

subplot (3,2,3), plot (Time, y1,'r')
hold on
subplot (3,2,3), plot (Time, Data_DT(:,3))
subplot (3,2,3), plot (Time, Touch(:,3),'k')
xlabel ('time[s]','FontSize',14)
ylabel (YL,'FontSize',14)
ylim([-0.5 0.5])
xlim([0 60])
title ('3rd Test w/ Threshold','FontSize',14)

subplot (3,2,4), plot (Time, y1,'r')
hold on
subplot (3,2,4), plot (Time, Data_DT(:,4))
subplot (3,2,4), plot (Time, Touch(:,4),'k')
xlabel ('time[s]','FontSize',14)
ylabel (YL,'FontSize',14)
ylim([-0.5 0.5])
xlim([0 60])
title ('4th Test w/ Threshold','FontSize',14)

subplot (3,2,5), plot (Time, y1,'r')
hold on
subplot (3,2,5), plot (Time, Data_DT(:,5))
subplot (3,2,5), plot (Time, Touch(:,5),'k')
xlabel ('time[s]','FontSize',14)
ylabel (YL,'FontSize',14)
ylim([-0.5 0.5])
xlim([0 60])

```

```
title ('5th Test w/ Threshold','FontSize',14)
```

```
%% New touch data
```

```
reply = input('New Data Update? Y/N [N]:','s');  
if isempty(reply)  
    reply = 'No updates!!!'
```

```
else  
    w=str2num(reply);  
    for q = 6:w  
        p=p+1;  
        X=['TouchTest',num2str(q),'.xlsx'];  
        [NUM,TXT,Data] = xlsread(X);  
        for i=1:N  
            Raw(i,1)=Data{i+1,2};  
        end  
        for j =1:4  
            m=M(j);  
            for i=1:N
```

```
if i<=m  
    Temp(i,j)=mean(Raw(1:i,1));  
elseif i>N-m  
    Temp(i,j)=mean(Raw(i:N,1));  
else  
    Temp(i,j)=mean(Raw(i-m:i+m,1));  
end
```

```
if i>1  
    DT (i,j)=Temp(i,j)-Temp(i-1,j);  
end  
Data_T(:,p)=Temp(:,4);  
Data_DT(:,p)=DT(:,4);  
end  
end  
New_Touch = zeros(N,1);  
A_touch=0;  
for j=1:N-10  
    if Data_DT(j,p)>=Th  
        for k=10:N-j  
            if mean(Data_DT(j:j+k,p))<=Th  
  
                break  
            else  
                New_Touch(j:j+k,1)=1;
```

```

        A_touch=1;
        if j>=N-10
            break
        end
    end
end
end
end
end

if A_touch ==0
    p=p-1;
    disp(['No touch detected for TouchTest ',num2str(q)])
else
    New_max=max(Data_DT(:,p));
    disp(['Maximum temperature change in 0.1s for new touch testes'])
    disp([New_max])
    MaxDT(p)=New_max;
    Mean_MaxDT = mean(MaxDT);
    disp(['Updated Mean of maximum temperature change in 0.1s touch testes = ',num2str(Mean_MaxDT)])
    Std_MaxDT = std(MaxDT);
    disp(['Updated Standard deviation of maximum temperature change in 0.1s for touch testes =',num2str(Std_MaxDT)])
    disp(['For 99% accuracy of normal distribution, We update Threshold = Mean - 2.57*Std' ])
    Th = Mean_MaxDT - 1.96 * Std_MaxDT;
    disp(['New Threshold = ', num2str(Th)])
    end
    %%
    YL=['Temperature[C', char(0176), ']'];
    figure (4)
    hold on
    subplot (2,1,1) ,plot (Time, Temp(:,1))
    xlabel ('time[s'],'FontSize',16)
    ylabel (YL,'FontSize',16)
    ylim([20 32])
    xlim([0 60])
    title ('New Data Temperature vs Time ','FontSize',16)

    figure (4)
    subplot (2,1,2), plot (Time, y1,'r')
    hold on
    subplot (2,1,2) ,plot (Time, DT(:,4))
    subplot (2,1,2), plot (Time, New_Touch(:,1),'k')
    hold off
    xlabel ('time[s'],'FontSize',16)
    ylabel (YL,'FontSize',16)
    ylim([-0.5 0.5])
    xlim([0 60])
    title ('New Data Temperature change in 0.1s ','FontSize',16)

```

```

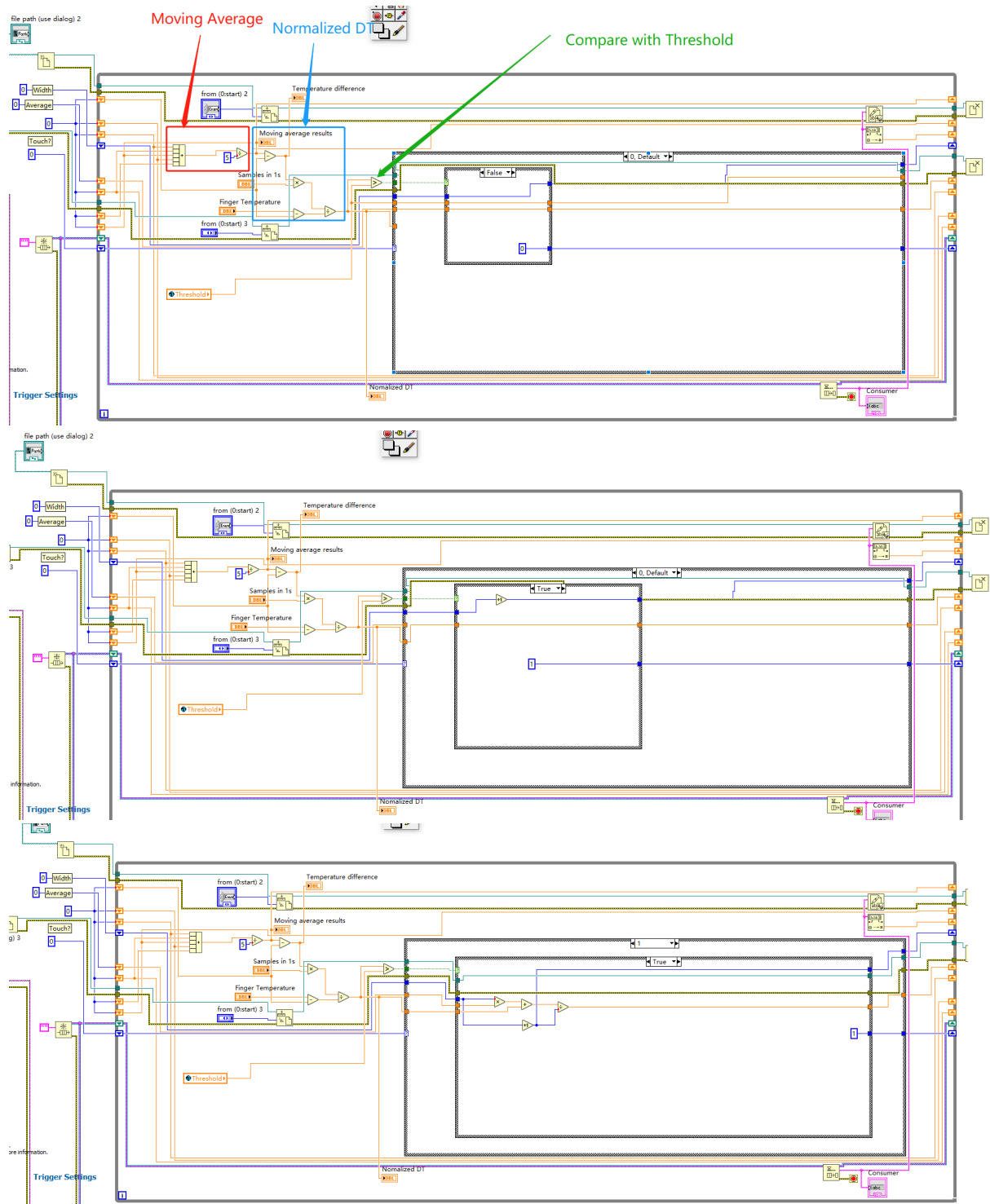
% New_max=max(Data_DT(:,p));
% disp(['Maximum temperature change in 0.1s for new touch testes'])
% disp([New_max])
% MaxDT(p)=New_max;
% Mean_MaxDT = mean(MaxDT);
% disp(['Updated Mean of maximum temperature change in 0.1s touch testes =
',num2str(Mean_MaxDT)])
% Std_MaxDT = std(MaxDT);
% disp(['Updated Standard deviation of maximum temperature change in 0.1s for touch testes
=',num2str(Std_MaxDT)])
% disp(['For 99% accuracy of normal distribution, We update Threshold = Mean - 2.57*Std' ])
% Th = Mean_MaxDT - 2.57 * Std_MaxDT;
% disp(['New Threshold = ', num2str(Th)])

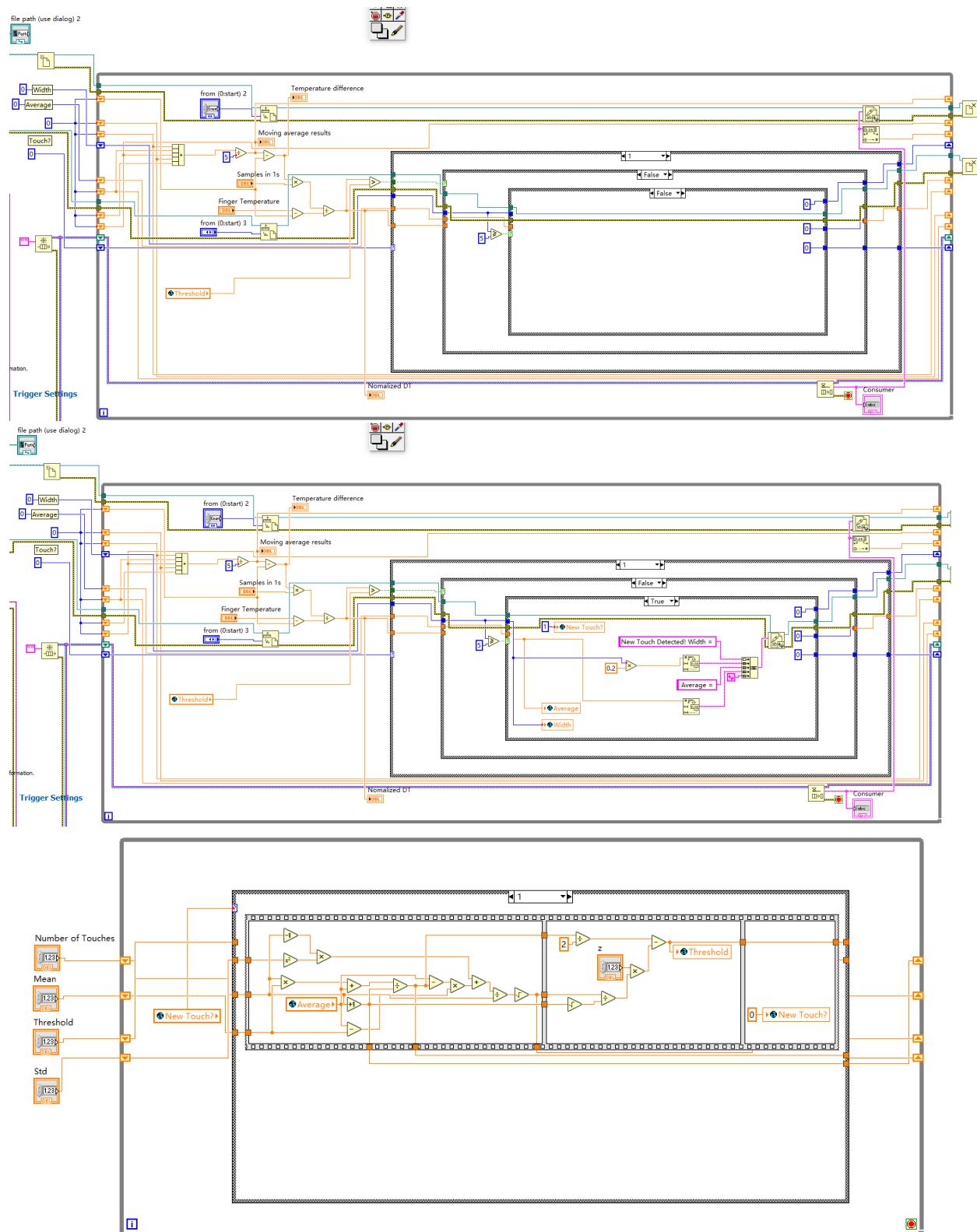
    end

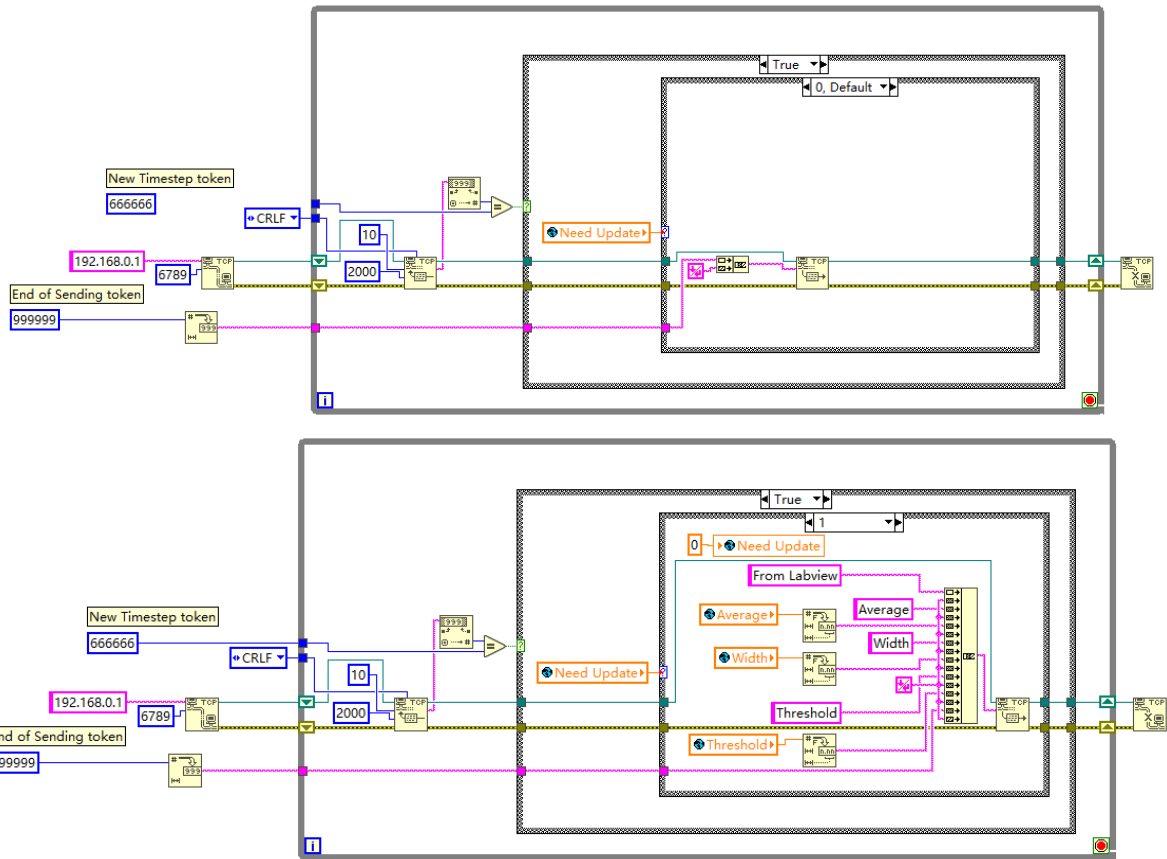
end

```

Appendix B. Temperature Data Collection (LabView)







Appendix C. PIR Motion Sensor Collection (BBB)

```
package org.webgme.guest.bbb;

import org.webgme.guest.bbb.rti.*;

import org.cpswt.config.FederateConfig;
import org.cpswt.config.FederateConfigParser;
import org.cpswt.hla.base.AdvanceTimeRequest;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import java.io.*;

// Define the BBB type of federate for the federation.

public class BBB extends BBBBase {
    private final static Logger log = LogManager.getLogger();

    private double currentTime = 0;

    public BBB(FederateConfig params) throws Exception {
        super(params);
    }

    private int Pinvalue(String x) throws Exception{
        String Path = "/sys/class/gpio/gpio" +x+ "/value";
        FileReader fr=new FileReader(Path);
        int i,v;
        i=fr.read();
        v= i -48;
        fr.close();
        return v;
    }

    private void SetValue(String x,String v) throws Exception{
        String Path = "/sys/class/gpio/gpio" +x+ "/value";
        FileWriter wr=new FileWriter(Path);
        wr.write(v);
        wr.close();
    }

    private int Display(String Switch,String Led) throws Exception{
```

```

int value = Pinvalue(Switch);
String HIGH = "1", LOW = "0";
if (value==1){
    SetValue(Led, HIGH);
}else{
    SetValue(Led, LOW);
}
return value;
}

String Pin1 ="69", Pin2 ="45", Pin3 ="23", Pin4 ="47", Pin5 ="27";
String Led1 ="68", Led2 ="44", Led3 ="26", Led4 ="46", Led5 ="65";
int value1,value2,value3,value4,value5;

private void execute() throws Exception {
    if(super.isLateJoiner()) {
        log.info("turning off time regulation (late joiner)");
        currentTime = super.getLBTS() - super.getLookAhead();
        super.disableTimeRegulation();
    }

    //////////////////////////////////////
    // TODO perform basic initialization below //
    //////////////////////////////////////

    AdvanceTimeRequest atr = new AdvanceTimeRequest(currentTime);
    putAdvanceTimeRequest(atr);

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToPopulate...");
        readyToPopulate();
        log.info("...synchronized on readyToPopulate");
    }

    //////////////////////////////////////
    // TODO perform initialization that depends on other federates below //
    //////////////////////////////////////

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToRun...");
        readyToRun();
        log.info("...synchronized on readyToRun");
    }

    startAdvanceTimeThread();
    log.info("started logical time progression");

    while (!exitCondition) {
        atr.requestSyncStart();

```

```

enteredTimeGrantedState();

value1 = Display(Pin1,Led1);
value2 = Display(Pin2,Led2);
value3 = Display(Pin3,Led3);
value4 = Display(Pin4,Led4);
value5 = Display(Pin5,Led5);

System.out.println("Current State: \n"+ value1 +"\n"+ value2 +"\n"+ value3 +"\n"+ value4 +"\n"+
value5);

```

```

////////////////////////////////////
// TODO send interactions that must be sent every logical //
// time step below                                     //
////////////////////////////////////

// Set the Switch1 interaction's.

Data Switch1 = create_Data();
Switch1.set_value(Double.valueOf(value1));
Switch1.set_varName( "Switch1" );
Switch1.sendInteraction(getLRC(), currentTime + getLookAhead());

// Set the Switch2 interaction's.

Data Switch2 = create_Data();
Switch2.set_value(Double.valueOf(value2));
Switch2.set_varName( "Switch2" );
Switch2.sendInteraction(getLRC(), currentTime + getLookAhead());

// Set the Switch3 interaction's.

Data Switch3 = create_Data();
Switch3.set_value(Double.valueOf(value3));
Switch3.set_varName( "Switch3" );
Switch3.sendInteraction(getLRC(), currentTime + getLookAhead());

// Set the Switch4 interaction's.

Data Switch4 = create_Data();
Switch4.set_value(Double.valueOf(value4));
Switch4.set_varName( "Switch4" );
Switch4.sendInteraction(getLRC(), currentTime + getLookAhead());

// Set the Switch5 interaction's.

Data Switch5 = create_Data();
Switch5.set_value(Double.valueOf(value5));

```

```

        Switch5.set_varName( "Switch5" );
        Switch5.sendInteraction(getLRC(), currentTime + getLookAhead());

        ////////////////////////////////////
        // TODO break here if ready to resign and break out of while loop //
        ////////////////////////////////////

        if (!exitCondition) {
            currentTime += super.getStepSize();
            AdvanceTimeRequest newATR =
                new AdvanceTimeRequest(currentTime);
            putAdvanceTimeRequest(newATR);
            atr.requestSyncEnd();
            atr = newATR;
        }
    }

    // call exitGracefully to shut down federate
    exitGracefully();

    ////////////////////////////////////
    // TODO Perform whatever cleanups are needed before exiting the app //
    ////////////////////////////////////
}

public static void main(String[] args) {
    try {
        FederateConfigParser federateConfigParser =
            new FederateConfigParser();
        FederateConfig federateConfig =
            federateConfigParser.parseArgs(args, FederateConfig.class);
        BBB federate =
            new BBB(federateConfig);
        federate.execute();
        log.info("Done.");
        System.exit(0);
    }
    catch (Exception e) {
        log.error(e);
        System.exit(1);
    }
}
}

```

Appendix D. Relative Time Event Sequence Format

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Arrays;

public class Detection {

    public static void main(String[] args) {
        String Date = "20191018";
        String csvFile = "/" + Date + ".csv";
        String line = "";
        String cvsSplitBy = ",";
        String header;

        int time, PIR1;
        double indoorHandle, outdoorHandle, tap, toilet;
        int indoorTouch, outdoorTouch, tapUsage, toiletUsage;
        int eventStartTime = 0;
        int inEvent = 0;
        int[][] EventData = new int[300][12];

        int relativeTime = 0;
        int lastTriggerTime = 0;

        int[] indoorTrigger = {0, 0, 0};
        int[] outdoorTrigger = {0, 0, 0};
        int[] tapTrigger = {0, 0, 0};
        int[] toiletTrigger = {0, 0, 0};
        int[] PIRTrigger = {0, 0, 0, 0};

        int indoorN = 0;
        int outdoorN = 0;
        int tapN = 0;
        int toiletN = 0;
        int PIRN = 0;

        int lastIndoorTouch = 0;
        int lastOutdoorTouch = 0;
        int lastTapUsage = 0;
        int lastToiletUsage = 0;
        int lastPIR = 0;

        int switch1 = 0;
        int switch2 = 0;
        int switch3 = 0;
```



```

int switch4 = 0;
int switch5 = 0;

int Event1, Event2, Event3, Event4, Event5;

String Lable;

try (BufferedReader br = new BufferedReader(new FileReader(csvFile))) {
    header = br.readLine();
    System.out.println(header);
    while ((line = br.readLine()) != null) {
        // for (int i=1;i<=5;i++){
        //     line = br.readLine();
        // use comma as separator
        String[] Data = line.split(csvSplitBy);
        //System.out.println(Arrays.toString(Data));
        time = Integer.parseInt(Data[1]);

        indoorHandle = Double.parseDouble(Data[2]);
        outdoorHandle = Double.parseDouble(Data[3]);
        tap = Double.parseDouble(Data[6]);
        toilet = Double.parseDouble(Data[7]);
        PIR1 = (int) Double.parseDouble(Data[13]);

        Event1 = (int)Double.parseDouble(Data[8]) - switch1;
        Event2 = (int)Double.parseDouble(Data[9]) - switch2;
        Event3 = (int)Double.parseDouble(Data[10]) - switch3;
        Event4 = (int)Double.parseDouble(Data[11]) - switch4;
        Event5 = (int)Double.parseDouble(Data[12]) - switch5;

        switch1 = (int)Double.parseDouble(Data[8]);
        switch2 = (int)Double.parseDouble(Data[9]);
        switch3 = (int)Double.parseDouble(Data[10]);
        switch4 = (int)Double.parseDouble(Data[11]);
        switch5 = (int)Double.parseDouble(Data[12]);

        //
        System.out.println("=====
        =====");
        // System.out.println("Date : "+ Date + " Time : "+ time);
        // System.out.println("Indoor Handle = "+ indoorHandle + " Outdoor Handle = "+
        outdoorHandle + " Tap = "+ tap + " Toilet = " + toilet + " PIR = " + PIR1 );
        // System.out.println("Switch 1 : " + switch1 +" | Switch 2 : " + switch2 +" | Switch 3 : " +
        switch3 +" | Switch 4 : " + switch4 +" | Switch 5 : " + switch5);
        // System.out.println("-----");

        //Convert to binary data

        if (indoorHandle>=0.015){

```

```

        indoorTouch = 1;
    }else{
        indoorTouch = 0;
    }
    if (outdoorHandle>=0.012){
        outdoorTouch = 1;
    }else{
        outdoorTouch = 0;
    }
    if (tap>=0.007){
        tapUsage = 1;
    }else{
        tapUsage = 0;
    }
    if (toilet>=0.007){
        toiletUsage = 1;
    }else{
        toiletUsage = 0;
    }

    // System.out.println("Indoor Touch : "+ indoorTouch + " Outdoor Touch : "+ outdoorTouch +
    " Tap Usage : "+ tapUsage + " Toilet Usage : " + toiletUsage + " Motion Near Door : " + PIR1 );
    //
    System.out.println("=====
    =====");

    //Event Detection
    if (inEvent == 0 && ( (indoorTouch == 1) || (outdoorTouch == 1) || (tapUsage == 1) ||
    (toiletUsage == 1) || (PIR1 == 1))){
        System.out.println("New Event Detected !!!" + "Date : "+ Date + " Time : "+ time);
        eventStartTime = time;

        inEvent = 1;
    }
    if (inEvent == 1){

        relativeTime = relativeTime + 1;

        EventData[relativeTime-1][0] = time;
        EventData[relativeTime-1][1] = relativeTime;
        EventData[relativeTime-1][2] = indoorTouch;
        EventData[relativeTime-1][3] = outdoorTouch;
        EventData[relativeTime-1][4] = tapUsage;
        EventData[relativeTime-1][5] = toiletUsage;
        EventData[relativeTime-1][6] = PIR1;
        EventData[relativeTime-1][7] = Event1;
        EventData[relativeTime-1][8] = Event2;
        EventData[relativeTime-1][9] = Event3;
        EventData[relativeTime-1][10] = Event4;

```

```

EventData[relativeTime-1][11] = Event5;

}

if (indoorTouch == 1 || outdoorTouch == 1 || tapUsage == 1 || toiletUsage == 1 || PIR1 == 1){
    lastTriggerTime = relativeTime;
}

if ((indoorTouch == 1) && (indoorN < 3) && (lastIndoorTouch != indoorTouch)){
    indoorTrigger[indoorN] = relativeTime;
    indoorN = indoorN + 1;
}

if ((outdoorTouch == 1) && (outdoorN < 3) && (lastOutdoorTouch != outdoorTouch)){
    outdoorTrigger[outdoorN] = relativeTime;
    outdoorN = outdoorN + 1;
}

if ((tapUsage == 1) && (tapN < 3) && (lastTapUsage != tapUsage)){
    tapTrigger[tapN] = relativeTime;
    tapN = tapN + 1;
}

if ((toiletUsage == 1) && (toiletN < 3) && (lastToiletUsage != toiletUsage)){
    toiletTrigger[toiletN] = relativeTime;
    toiletN = toiletN + 1;
}

if ((PIR1 == 1) && (lastPIR != PIR1)){
    PIRTrigger[3] = relativeTime;
    if (PIRN < 3){
        PIRTrigger[PIRN] = relativeTime;
        PIRN = PIRN + 1;
    }
}

if (Event1 != 0){
    if (Event1 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 1 is " + Lable + " at " + time + " Lable = " + Event1 );
}

if (Event2 != 0){
    if (Event2 == 1){
        Lable = "Entering";
    } else {

```

```

        Lable = "Leaving";
    }
    System.out.println("User 2 is " + Lable + " at "+ time +" Lable = "+ Event2 );
}

if (Event3 != 0){
    if (Event3 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 3 is " + Lable + " at "+ time +" Lable = "+ Event3 );
}

if (Event4 != 0){
    if (Event4 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 4 is " + Lable + " at "+ time +" Lable = "+ Event4 );
}

if (Event5 != 0){
    if (Event5 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 5 is " + Lable + " at "+ time +" Lable = "+ Event5 );
}

lastIndoorTouch = indoorTouch;
lastOutdoorTouch = outdoorTouch;
lastTapUsage = tapUsage;
lastToiletUsage = toiletUsage;
lastPIR = PIR1;

if ((relativeTime - lastTriggerTime >= 30) ^ (relativeTime >= 299)){

    inEvent = 0;

    System.out.println("=====
=====");

```

```

System.out.println("time\t\trelativeTime\tindoorTouch\toutdoorTouch\ttapUsage\ttoiletUsage\tPIR1\t\tU
ser1\t\tUser2\t\tUser3\t\tUser4\t\tUser5");
    for (int i=0; i<=lastTriggerTime; i++){
        for(int j = 0; j < EventData[i].length; j++){
            System.out.print(EventData[i][j]+"\\t\\t");
        }
        System.out.print("\\n");
    }

    System.out.println("-----");
    System.out.println("Event Ended !!! " + "Date : "+ Date + "   Event Start Time: "+
eventStartTime);
    System.out.println("Indoor Touch at : " + Arrays.toString(indoorTrigger));
    System.out.println("Outdoor Touch at : " + Arrays.toString(outdoorTrigger));
    System.out.println("Tap Usage at : " + Arrays.toString(tapTrigger));
    System.out.println("Toilet Usage at : " + Arrays.toString(toiletTrigger));
    System.out.println("Motion Near Door at : " + Arrays.toString(PIRTrigger));

System.out.println("=====
=====");

    relativeTime = 0;
    lastTriggerTime = 0;

    lastIndoorTouch = 0;
    lastOutdoorTouch = 0;
    lastTapUsage = 0;
    lastToiletUsage = 0;
    lastPIR = 0;

    Arrays.fill(indoorTrigger, 0);
    Arrays.fill(outdoorTrigger, 0);
    Arrays.fill(tapTrigger, 0);
    Arrays.fill(toiletTrigger, 0);
    Arrays.fill(PIRTrigger, 0);

    indoorN = 0;
    outdoorN = 0;
    tapN = 0;
    toiletN = 0;
    PIRN = 0;

    }

    }

} catch (IOException e) {

```

```
        e.printStackTrace();  
    }  
}  
}
```

Appendix E. Machine Learning Code

```
###
import numpy as np
import pandas as pd
import glob
import sklearn
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pylab import rcParams

from joblib import dump,load

#####
# read all csv into pandas dataframe #
#####
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC, NuSVC

li = []
paths = sorted(glob.glob('.\\machine_learning_input_without_tap_and_toilet.csv'))
# paths = sorted(glob.glob('.\\RightBranch.csv'))

for path in paths:
    li.append(pd.read_csv(path))

df = pd.concat(li)
#1. df.shape is to get the size of lines
print('The RAW data shape is: ', df.shape)
df_2 = df.iloc[23:24]
print(df_2)
print('-----')
df_3 = df.drop([12, 13, 14, 15, 16, 22, 21, 23, 24])
print(df_3)
# df_3 = df_3.drop([1,2,4,5,6,7,8,9,10,11,17,18,19,20])
# print(df_3)
print('-----')

df_1 = df_3.T
train_list = np.array(df_1) #np.ndarray()
train_data = train_list.tolist() #list
del train_data[0]
```

```

#print(train_data)
train_data_array = np.array(train_data)
train_data_array = train_data_array.astype(np.float64)
# print(train_data_array)

train_list_1 = np.array(df_2) #np.ndarray()
train_label = train_list_1.tolist() #list
train_label = train_label[0]
del train_label[0]
#print(train_label)
train_label_array = np.array(train_label)
train_label_array = train_label_array.astype(np.float64)
# print(train_label_array)

X_train, X_test, y_train, y_test = train_test_split(
    train_data_array,
    train_label_array,
    test_size=0.2,
    random_state=42
)
print('training sets\' shapes: ', X_train.shape, y_train.shape)
print('testing sets\' shapes: ', X_test.shape, y_test.shape)

classifiers = [
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    KNeighborsClassifier(),
    SVC()
]

y_preds = []

###
# train and validate
###

y_preds = []
clf = RandomForestClassifier()
Tree = clf.fit(X_train, y_train)
name = clf.__class__.__name__
y_pred = clf.predict(X_test)
y_preds.append(y_pred)
print(name)
print('Accuracy: ', sklearn.metrics.accuracy_score( y_test, y_pred ) )
print( sklearn.metrics.classification_report( y_test, y_pred ) )
print( sklearn.metrics.confusion_matrix( y_test, y_pred ) )
importance = clf.feature_importances_
print(importance)

```



```

print('-----')

dump(clf,'RandomForest_ALL.joblib')

y_preds = []
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)
name = clf.__class__.__name__
y_pred = clf.predict(X_test)
y_preds.append(y_pred)
print(name)
print('Accuracy: ', sklearn.metrics.accuracy_score( y_test, y_pred ) )
print( sklearn.metrics.classification_report( y_test, y_pred ) )
print( sklearn.metrics.confusion_matrix( y_test, y_pred ) )

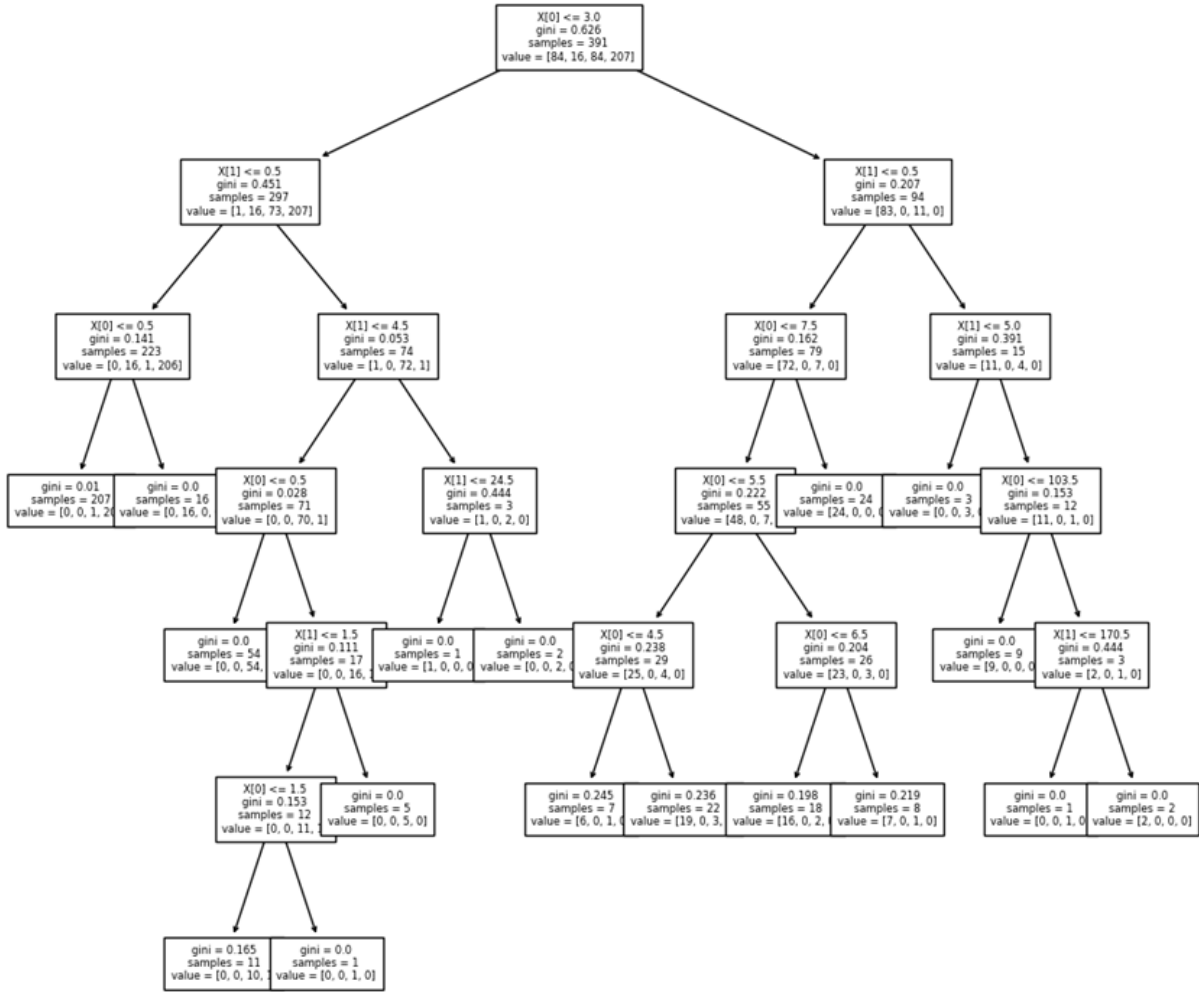
y_preds = []
clf = KNeighborsClassifier()
clf.fit(X_train, y_train)
name = clf.__class__.__name__
y_pred = clf.predict(X_test)
y_preds.append(y_pred)
print(name)
print('Accuracy: ', sklearn.metrics.accuracy_score( y_test, y_pred ) )
print( sklearn.metrics.classification_report( y_test, y_pred ) )
print( sklearn.metrics.confusion_matrix( y_test, y_pred ) )

y_preds = []
clf = SVC(gamma='auto')
clf.fit(X_train, y_train)
name = clf.__class__.__name__
y_pred = clf.predict(X_test)
y_preds.append(y_pred)
print(name)
print('Accuracy: ', sklearn.metrics.accuracy_score( y_test, y_pred ) )
print( sklearn.metrics.classification_report( y_test, y_pred ) )
print( sklearn.metrics.confusion_matrix( y_test, y_pred ) )

# plt.figure(figsize=(12,11))
# sklearn.tree.plot_tree(clf,fontsize=6)
# plt.show()

```

Appendix F. Decision Tree



Appendix G. Detailed Result: Without Human Activity Layer

Decision Tree				
Confusion Matrix		Predicted		
		Vacant	Occupied	
Ground truth	Vacant	529914	1284	
	Occupied	87994	2900	
Performance		Vacant	Occupied	Macro-average
Precision		0.86	0.69	0.78
Recall		1.00	0.03	0.51
F1-score		0.62		
Accuracy		0.86		

Random Forest				
Confusion Matrix		Predicted		
		Vacant	Occupied	
Ground truth	Vacant	530170	1028	
	Occupied	88007	2887	
Performance		Vacant	Occupied	Macro-average
Precision		0.86	0.74	0.80
Recall		1.00	0.03	0.51
F1-score		0.63		
Accuracy		0.86		

K-Nearest Neighbors				
Confusion Matrix		Predicted		
		Vacant	Occupied	
Ground truth	Vacant	528366	2832	
	Occupied	87232	3662	
Performance		Vacant	Occupied	Macro-average
Precision		0.86	0.56	0.71
Recall		0.99	0.04	0.52
F1-score		0.60		
Accuracy		0.86		

SVM				
Confusion Matrix		Predicted		
		Vacant	Occupied	
Ground truth	Vacant	598592	2096	
	Occupied	89285	1225	
Performance		Vacant	Occupied	Macro-average
Precision		0.87	0.37	0.62
Recall		1.00	0.01	0.51
F1-score		0.56		
Accuracy		0.87		

Appendix H. Detailed Result: With Human Activity Layer

Decision Tree					
Confusion Matrix		Predicted			
		Leaving	Entering	No Change	
Ground truth	Leaving	14	1	1	
	Entering	1	15	0	
	No Change	0	1	64	
Performance		Leaving	Entering	No Change	Macro-average
Precision		0.93	0.88	0.98	0.93
Recall		0.88	0.94	0.98	0.93
F1-score		0.93			
Accuracy		0.96			

Random Forest					
Confusion Matrix		Predicted			
		Leaving	Entering	No Change	
Ground truth	Leaving	15	1	0	
	Entering	0	16	0	
	No Change	0	0	65	
Performance		Leaving	Entering	No Change	Macro-average
Precision		1.00	0.94	1.00	0.98
Recall		0.94	1.00	1.00	0.98
F1-score		0.98			
Accuracy		0.99			

K-Nearest Neighbors					
Confusion Matrix		Predicted			
		Leaving	Entering	No Change	
Ground truth	Leaving	16	0	0	
	Entering	1	12	3	
	No Change	0	1	64	
Performance		Leaving	Entering	No Change	Macro-average
Precision		0.94	0.92	0.96	0.94
Recall		1.00	0.75	0.98	0.91
F1-score		0.93			
Accuracy		0.95			

SVC					
Confusion Matrix		Predicted			
		Leaving	Entering	No Change	
Ground truth	Leaving	9	0	7	
	Entering	0	13	3	
	No Change	0	0	65	
Performance		Leaving	Entering	No Change	Macro-average
Precision		1.00	1.00	0.87	0.96
Recall		0.56	0.81	1.00	0.79
F1-score		0.87			
Accuracy		0.90			

Appendix I. Cost Estimation of a Sensor System

The table below shows the prototype cost estimation of a one-entrance and one-bathroom sensor system.

Bill of materials				
	Type of sensor	Unit Price [\$]	Quantity	Value [\$]
Door Package	Thermistor	\$ 0.05	2	\$ 0.10
	PIR	\$ 2.00	1	\$ 2.00
	SoC(Blue tooth)	\$ 30.00	1	\$ 30.00
Bathroom package	Thermistor	\$ 0.05	2	\$ 0.10
	SoC(Blue tooth)	\$ 30.00	1	\$ 30.00
Data Hub	SoC(Blue tooth + Wifi)	\$ 30.00	1	\$ 30.00
			Total	\$ 92.20

Appendix J. TCP/IP Host in UCEF

```
package org.webgme.guest.socket;

import org.webgme.guest.socket.rti.*;

import org.cpswt.config.FederateConfig;
import org.cpswt.config.FederateConfigParser;
import org.cpswt.hla.InteractionRoot;
import org.cpswt.hla.base.AdvanceTimeRequest;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import java.io.*;
import java.net.*;

// Define the Socket type of federate for the federation.

public class Socket extends SocketBase {
    private final static Logger log = LogManager.getLogger();

    private double currentTime = 0;

    String eGSH=null, eGSC=null, setName=null, ePeople=null;
    boolean empty=true;

    public Socket(FederateConfig params) throws Exception {
        super(params);
    }

    private void checkReceivedSubscriptions() {
        InteractionRoot interaction = null;
        while ((interaction = getNextInteractionNoWait()) != null) {
            if (interaction instanceof SendEP) {
                handleInteractionClass((SendEP) interaction);
            }
            else {
                log.debug("unhandled interaction: {}", interaction.getClassName());
            }
        }
    }

    private void execute() throws Exception {
        if(super.isLateJoiner()) {
            log.info("turning off time regulation (late joiner)");
        }
    }
}
```

```

        currentTime = super.getLBTS() - super.getLookAhead();
        super.disableTimeRegulation();
    }

    //////////////////////////////////////////////////
    // TODO perform basic initialization below //
    //////////////////////////////////////////////////

    //////////////////////////////////////////////////Socket
    InetAddress addr = InetAddress.getByName("192.168.1.141"); // the address needs to be
changed
    ServerSocket welcomeSocket = new ServerSocket(6789, 50, addr); // 6789 is port number. Can
be changed
    java.net.Socket connectionSocket = welcomeSocket.accept(); // initial connection will be made at
this point
    System.out.println("connection successful");
    log.info("connection successful");

    InputStreamReader inFromClient = new InputStreamReader(connectionSocket.getInputStream());
    BufferedReader buffDummy = new BufferedReader(inFromClient);
    DataOutputStream outToClient = new DataOutputStream(connectionSocket.getOutputStream());
    //////////////////////////////////////////////////Socket

    AdvanceTimeRequest atr = new AdvanceTimeRequest(currentTime);
    putAdvanceTimeRequest(atr);

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToPopulate...");
        readyToPopulate();
        log.info("...synchronized on readyToPopulate");
    }

    //////////////////////////////////////////////////
    // TODO perform initialization that depends on other federates below //
    //////////////////////////////////////////////////

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToRun...");
        readyToRun();
        log.info("...synchronized on readyToRun");
    }

    startAdvanceTimeThread();
    log.info("started logical time progression");

    String header, time="0", varname, value;
    double varValue;

```

```

while (!exitCondition) {
    atr.requestSyncStart();
    enteredTimeGrantedState();

    checkReceivedSubscriptions();

    if((header = buffDummy.readLine()).equals("TERMINATE")){
        exitCondition = true;
    }

    time = buffDummy.readLine();
    System.out.println("in loop header=" + header + " t=" + time);

    while(!(varname = buffDummy.readLine()).isEmpty()) {
        value = buffDummy.readLine();
        System.out.println("Received: " + varname + " as " + value);
        varValue = Double.parseDouble(value);

        if(varname.equals("epSendOutdoorAirTemp")){
            ReceiveEP sendOutTemp = create_ReceiveEP();

            sendOutTemp.set_varName( varname );
            sendOutTemp.set_value( varValue );

            log.info("Sent Out Temp interaction as {} = {}", varname , varValue);

            sendOutTemp.sendInteraction(getLRC(), currentTime + getLookAhead());
        }

        if(varname.equals("epSendZoneMeanAirTemp")){
            ReceiveEP sendZoneTemp = create_ReceiveEP();

            sendZoneTemp.set_varName( varname );
            sendZoneTemp.set_value( varValue );

            log.info("Sent Zone Mean Temp interaction as {} = {}", varname , varValue);

            sendZoneTemp.sendInteraction(getLRC(), currentTime + getLookAhead());
        }

        if(varname.equals("epSendZoneHumidity")){
            ReceiveEP sendRH = create_ReceiveEP();

            sendRH.set_varName( varname );

```

```

        sendRH.set_value( varValue );

        log.info("Sent Humidity interaction as {} = {}" , varname , varValue);

        sendRH.sendInteraction(getLRC(), currentTime + getLookAhead());
    }
    //////////////////////////////////////
    // TODO send interactions that must be sent every logical //
    // time step below                                     //
    //////////////////////////////////////

    // Set the interaction's parameters.
    //
    // ReceiveEP vReceiveEP = create_ReceiveEP();
    // vReceiveEP.set_actualLogicalGenerationTime( < YOUR VALUE HERE > );
    // vReceiveEP.set_federateFilter( < YOUR VALUE HERE > );
    // vReceiveEP.set_originFed( < YOUR VALUE HERE > );
    // vReceiveEP.set_sourceFed( < YOUR VALUE HERE > );
    // vReceiveEP.set_unit( < YOUR VALUE HERE > );
    // vReceiveEP.set_value( < YOUR VALUE HERE > );
    // vReceiveEP.set_varName( < YOUR VALUE HERE > );
    // vReceiveEP.sendInteraction(getLRC(), currentTime + getLookAhead());

    }
    if (empty==true) {
        outToClient.writeBytes("NOUPDATE\r\n\r\n");
    } else {
        outToClient.writeBytes("SET\r\n" + time + "\r\n" + "epGetStartCooling\r\n" + eGSC + "\r\n" +
"epGetStartHeating\r\n" + eGSH + "\r\n" + "\r\n");
        System.out.println("SET\r\n" + time + "\r\n" + "epGetStartCooling\r\n" + eGSC + "\r\n" +
"epGetStartHeating\r\n" + eGSH + "\r\n" + "\r\n");
    }
    outToClient.flush();

    //////////////////////////////////////
    // TODO break here if ready to resign and break out of while loop //
    //////////////////////////////////////

    if (!exitCondition) {
        currentTime += super.getStepSize();
        AdvanceTimeRequest newATR =
            new AdvanceTimeRequest(currentTime);
        putAdvanceTimeRequest(newATR);
        atr.requestSyncEnd();
        atr = newATR;
    }
}

// call exitGracefully to shut down federate

```

```

        exitGracefully();

        //////////////////////////////////////
        // TODO Perform whatever cleanups are needed before exiting the app //
        //////////////////////////////////////
    }

    private void handleInteractionClass(SendEP interaction) {
        //////////////////////////////////////
        // TODO implement how to handle reception of the interaction //
        //////////////////////////////////////

        double value = 20;
        empty = false;
        setName = interaction.get_varName();
        System.out.println("Received Data " + setName);
        if(setName.equals("epGetStartHeating")){
            value = interaction.get_value();
            eGSH = String.valueOf(value);
            System.out.println("Received Heating setpoint interaction as" + setName + eGSH);
            log.info("Received Heating setpoint interaction as {} = {}" , setName , eGSH);
        }
        if(setName.equals("epGetStartCooling")){
            value = interaction.get_value();
            eGSC = String.valueOf(value);
            System.out.println("Received Heating setpoint interaction as" + setName + eGSC);
            log.info("Received Cooling setpoint interaction as {} = {}" , setName , eGSC);
        }
        if(setName.equals("epGetPeople")){
            value = interaction.get_value();
            ePeople = String.valueOf(value);
            System.out.println("Received Heating setpoint interaction as" + setName + ePeople);
            log.info("Received People interaction as {} = {}" , setName , ePeople);
        }
    }

}

public static void main(String[] args) {
    try {
        FederateConfigParser federateConfigParser =
            new FederateConfigParser();
        FederateConfig federateConfig =
            federateConfigParser.parseArgs(args, FederateConfig.class);
        Socket federate =
            new Socket(federateConfig);
    }
}

```



```
        federate.execute();
        log.info("Done.");
        System.exit(0);
    }
    catch (Exception e) {
        log.error(e);
        System.exit(1);
    }
}
}
```

Appendix K. Data Fusion Code in UCEF

```
package org.webgme.guest.database;

import org.webgme.guest.database.rti.*;

import org.cpswt.config.FederateConfig;
import org.cpswt.config.FederateConfigParser;
import org.cpswt.hla.InteractionRoot;
import org.cpswt.hla.base.AdvanceTimeRequest;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import java.util.Arrays;
// Define the Database type of federate for the federation.

public class Database extends DatabaseBase {
    private final static Logger log = LogManager.getLogger();

    private double currentTime = 0;

    String Date = "20191018";

    int time, PIR1;
    double indoorHandle, outdoorHandle, tap, toilet;
    int indoorTouch, outdoorTouch, tapUsage, toiletUsage;
    int eventStartTime = 0;
    int inEvent = 0;
    int[][] EventData = new int[300][12];

    int relativeTime = 0;
    int lastTriggerTime = 0;

    int[] indoorTrigger = {0, 0, 0};
    int[] outdoorTrigger = {0, 0, 0};
    int[] tapTrigger = {0, 0, 0};
    int[] toiletTrigger = {0, 0, 0};
    int[] PIRTrigger = {0, 0, 0, 0};

    int indoorN = 0;
    int outdoorN = 0;
    int tapN = 0;
    int toiletN = 0;
    int PIRN = 0;

    int lastIndoorTouch = 0;
    int lastOutdoorTouch = 0;
```

```

int lastTapUsage = 0;
int lastToiletUsage = 0;
int lastPIR = 0;

int switch1 = 0;
int switch2 = 0;
int switch3 = 0;
int switch4 = 0;
int switch5 = 0;

int lastSwitch1 = 0;
int lastSwitch2 = 0;
int lastSwitch3 = 0;
int lastSwitch4 = 0;
int lastSwitch5 = 0;

int Event1, Event2, Event3, Event4, Event5;

String Lable;
int groundTruth;

String varName;
double varValue;

public Database(FederateConfig params) throws Exception {
    super(params);
}

private void checkReceivedSubscriptions() {
    InteractionRoot interaction = null;
    while ((interaction = getNextInteractionNoWait()) != null) {
        if (interaction instanceof RawData) {
            handleInteractionClass((RawData) interaction);
        }
        else {
            log.debug("unhandled interaction: {}", interaction.getClassName());
        }
    }
}

private void execute() throws Exception {
    if(super.isLateJoiner()) {
        log.info("turning off time regulation (late joiner)");
        currentTime = super.getLBTS() - super.getLookAhead();
        super.disableTimeRegulation();
    }

    //////////////////////////////////////
    // TODO perform basic initialization below //

```

```
////////////////////////////////////
```

```
AdvanceTimeRequest atr = new AdvanceTimeRequest(currentTime);  
putAdvanceTimeRequest(atr);
```

```
if(!super.isLateJoiner()) {  
    log.info("waiting on readyToPopulate...");  
    readyToPopulate();  
    log.info("...synchronized on readyToPopulate");  
}
```

```
////////////////////////////////////
```

```
// TODO perform initialization that depends on other federates below //  
////////////////////////////////////
```

```
if(!super.isLateJoiner()) {  
    log.info("waiting on readyToRun...");  
    readyToRun();  
    log.info("...synchronized on readyToRun");  
}
```

```
startAdvanceTimeThread();  
log.info("started logical time progression");
```

```
while (!exitCondition) {  
    atr.requestSyncStart();  
    enteredTimeGrantedState();
```

```
////////////////////////////////////
```

```
// TODO send interactions that must be sent every logical //  
// time step below //
```

```
////////////////////////////////////
```

```
// Set the interaction's parameters.
```

```
//
```

```
// GroundTruth vGroundTruth = create_GroundTruth();  
// vGroundTruth.set_Event( < YOUR VALUE HERE > );  
// vGroundTruth.set_Occupant( < YOUR VALUE HERE > );  
// vGroundTruth.set_Time( < YOUR VALUE HERE > );  
// vGroundTruth.set_actualLogicalGenerationTime( < YOUR VALUE HERE > );  
// vGroundTruth.set_federateFilter( < YOUR VALUE HERE > );  
// vGroundTruth.set_originFed( < YOUR VALUE HERE > );  
// vGroundTruth.set_sourceFed( < YOUR VALUE HERE > );  
// vGroundTruth.sendInteraction(getLRC(), currentTime + getLookAhead());  
// FixFormatData vFixFormatData = create_FixFormatData();  
// vFixFormatData.set_1stIndoorTouch( < YOUR VALUE HERE > );  
// vFixFormatData.set_1stOutdoorTouch( < YOUR VALUE HERE > );
```

```

// vFixFormatData.set_1stPIR( < YOUR VALUE HERE > );
// vFixFormatData.set_1stTapUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_1stToiletUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_2ndIndoorTouch( < YOUR VALUE HERE > );
// vFixFormatData.set_2ndOutdoorTouch( < YOUR VALUE HERE > );
// vFixFormatData.set_2ndPIR( < YOUR VALUE HERE > );
// vFixFormatData.set_2ndTapUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_2ndToiletUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_3rdIndoorTouch( < YOUR VALUE HERE > );
// vFixFormatData.set_3rdOutdoorTouch( < YOUR VALUE HERE > );
// vFixFormatData.set_3rdPIR( < YOUR VALUE HERE > );
// vFixFormatData.set_3rdTapUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_3rdToiletUsage( < YOUR VALUE HERE > );
// vFixFormatData.set_Date( < YOUR VALUE HERE > );
// vFixFormatData.set_LastPIR( < YOUR VALUE HERE > );
// vFixFormatData.set_StartTime( < YOUR VALUE HERE > );
// vFixFormatData.set_actualLogicalGenerationTime( < YOUR VALUE HERE > );
// vFixFormatData.set_federateFilter( < YOUR VALUE HERE > );
// vFixFormatData.set_originFed( < YOUR VALUE HERE > );
// vFixFormatData.set_sourceFed( < YOUR VALUE HERE > );
// vFixFormatData.sendInteraction(getLRC(), currentTime + getLookAhead());

```

```

checkReceivedSubscriptions();

```

```

Event1 = (int)(lastSwitch1 - switch1);
Event2 = (int)(lastSwitch2 - switch2);
Event3 = (int)(lastSwitch3 - switch3);
Event4 = (int)(lastSwitch4 - switch4);
Event5 = (int)(lastSwitch5 - switch5);

```

```

lastSwitch1 = switch1;
lastSwitch2 = switch2;
lastSwitch3 = switch3;
lastSwitch4 = switch4;
lastSwitch5 = switch5;

```

```

//Convert to binary data

```

```

if (indoorHandle>=0.015){
    indoorTouch = 1;
}else{
    indoorTouch = 0;
}
if (outdoorHandle>=0.012){
    outdoorTouch = 1;
}else{
    outdoorTouch = 0;
}
if (tap>=0.007){

```

```

        tapUsage = 1;
    }else{
        tapUsage = 0;
    }
    if (toilet>=0.007){
        toiletUsage = 1;
    }else{
        toiletUsage = 0;
    }

    if (inEvent == 0 && ( (indoorTouch == 1) || (outdoorTouch == 1) || (tapUsage == 1) || (toiletUsage
== 1) || (PIR1 == 1))){
        System.out.println("New Event Detected !!!" + "Date : "+ Date + " Time : "+ time);
        eventStartTime = time;

        inEvent = 1;
    }
    if (inEvent == 1){

        relativeTime = relativeTime + 1;

        EventData[relativeTime-1][0] = time;
        EventData[relativeTime-1][1] = relativeTime;
        EventData[relativeTime-1][2] = indoorTouch;
        EventData[relativeTime-1][3] = outdoorTouch;
        EventData[relativeTime-1][4] = tapUsage;
        EventData[relativeTime-1][5] = toiletUsage;
        EventData[relativeTime-1][6] = PIR1;
        EventData[relativeTime-1][7] = Event1;
        EventData[relativeTime-1][8] = Event2;
        EventData[relativeTime-1][9] = Event3;
        EventData[relativeTime-1][10] = Event4;
        EventData[relativeTime-1][11] = Event5;

    }

    if (indoorTouch == 1 || outdoorTouch == 1 || tapUsage == 1 || toiletUsage == 1 || PIR1 == 1){
        lastTriggerTime = relativeTime;
    }

    if ((indoorTouch == 1) && (indoorN < 3) && (lastIndoorTouch != indoorTouch)){
        indoorTrigger[indoorN] = relativeTime;
        indoorN = indoorN + 1;
    }

    if ((outdoorTouch == 1) && (outdoorN < 3) && (lastOutdoorTouch != outdoorTouch)){
        outdoorTrigger[outdoorN] = relativeTime;
        outdoorN = outdoorN + 1;
    }

```

```

if ((tapUsage == 1) && (tapN < 3) && (lastTapUsage != tapUsage)){
    tapTrigger[tapN] = relativeTime;
    tapN = tapN + 1;
}

if ((toiletUsage == 1) && (toiletN < 3) && (lastToiletUsage != toiletUsage)){
    toiletTrigger[toiletN] = relativeTime;
    toiletN = toiletN + 1;
}

if ((PIR1 == 1) && (lastPIR != PIR1)){
    PIRTrigger[3] = relativeTime;
    if (PIRN < 3 ){
        PIRTrigger[PIRN] = relativeTime;
        PIRN = PIRN + 1;
    }
}

if (Event1 != 0){
    if (Event1 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 1 is " + Lable + " at "+ time +" Lable = "+ Event1 );
}

if (Event2 != 0){
    if (Event2 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 2 is " + Lable + " at "+ time +" Lable = "+ Event2 );
}

if (Event3 != 0){
    if (Event3 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 3 is " + Lable + " at "+ time +" Lable = "+ Event3 );
}

if (Event4 != 0){
    if (Event4 == 1){
        Lable = "Entering";
    }
}

```

```

    } else {
        Lable = "Leaving";
    }
    System.out.println("User 4 is " + Lable + " at "+ time +" Lable = "+ Event4 );
}

if (Event5 != 0){
    if (Event5 == 1){
        Lable = "Entering";
    } else {
        Lable = "Leaving";
    }
    System.out.println("User 5 is " + Lable + " at "+ time +" Lable = "+ Event5 );
}

lastIndoorTouch = indoorTouch;
lastOutdoorTouch = outdoorTouch;
lastTapUsage = tapUsage;
lastToiletUsage = toiletUsage;
lastPIR = PIR1;

if ((relativeTime - lastTriggerTime >= 30) ^ (relativeTime >= 299)){

    inEvent = 0;

    System.out.println("=====
=====");
    System.out.println("time\t relativeTime\t indoorTouch\t outdoorTouch\t tapUsage\t
toiletUsage\t PIR1\t User1\t User2\t User3\t User4\t User5\t");
    for (int i=0; i<=lastTriggerTime; i++){
        for(int j = 0; j < EventData[i].length; j++){
            System.out.print(EventData[i][j]+"");
        }
        System.out.print("\n");
    }

    System.out.println("-----");
    System.out.println("Event Ended !!! " + "Date : "+ Date + " Event Start Time: "+
eventStartTime);
    System.out.println("Indoor Touch at : " + Arrays.toString(indoorTrigger));
    System.out.println("Outdoor Touch at : " + Arrays.toString(outdoorTrigger));
    System.out.println("Tap Usage at : " + Arrays.toString(tapTrigger));
    System.out.println("Toilet Usage at : " + Arrays.toString(toiletTrigger));
    System.out.println("Motion Near Door at : " + Arrays.toString(PIRTrigger));

```



```

System.out.println("=====
=====");

    relativeTime = 0;
    lastTriggerTime = 0;

    lastIndoorTouch = 0;
    lastOutdoorTouch = 0;
    lastTapUsage = 0;
    lastToiletUsage = 0;
    lastPIR = 0;

    Arrays.fill(indoorTrigger, 0);
    Arrays.fill(outdoorTrigger, 0);
    Arrays.fill(tapTrigger, 0);
    Arrays.fill(toiletTrigger, 0);
    Arrays.fill(PIRTrigger, 0);

    indoorN = 0;
    outdoorN = 0;
    tapN = 0;
    toiletN = 0;
    PIRN = 0;

}

////////////////////////////////////
// TODO break here if ready to resign and break out of while loop //
////////////////////////////////////

if (!exitCondition) {
    currentTime += super.getStepSize();
    AdvanceTimeRequest newATR =
        new AdvanceTimeRequest(currentTime);
    putAdvanceTimeRequest(newATR);
    atr.requestSyncEnd();
    atr = newATR;
}
}

// call exitGracefully to shut down federate
exitGracefully();

////////////////////////////////////
// TODO Perform whatever cleanups are needed before exiting the app //
////////////////////////////////////
}

```

```

private void handleInteractionClass(RawData interaction) {
    //////////////////////////////////////
    // TODO implement how to handle reception of the interaction //
    //////////////////////////////////////
    varName = interaction.get_varName();
    varValue = interaction.get_varValue();

    //////////////////////////////////////
    // Receive time
    if(varName.equals("time")){
        time = (int)varValue;
    }
    //////////////////////////////////////

    //////////////////////////////////////
    // Receive indoorHandle
    if(varName.equals("indoorHandle")){
        indoorHandle = varValue;
    }
    //////////////////////////////////////

    //////////////////////////////////////
    // Receive outdoorHandle
    if(varName.equals("outdoorHandle")){
        outdoorHandle = varValue;
    }
    //////////////////////////////////////

    //////////////////////////////////////
    // Receive tap
    if(varName.equals("tap")){
        tap = varValue;
    }
    //////////////////////////////////////

    //////////////////////////////////////
    // Receive toilet
    if(varName.equals("toilet")){
        toilet = varValue;
    }
    //////////////////////////////////////

    //////////////////////////////////////
    // Receive PIR1
    if(varName.equals("PIR1")){
        PIR1 = (int)varValue;
    }
    //////////////////////////////////////
}

```

```

////////////////////////////////////
// Receive switch1
if(varName.equals("switch1")){
    switch1 = (int)varValue;
}
////////////////////////////////////

////////////////////////////////////
// Receive switch2
if(varName.equals("switch2")){
    switch2 = (int)varValue;
}
////////////////////////////////////

////////////////////////////////////
// Receive switch3
if(varName.equals("switch3")){
    switch3 = (int)varValue;
}
////////////////////////////////////

////////////////////////////////////
// Receive switch4
if(varName.equals("switch4")){
    switch4 = (int)varValue;
}
////////////////////////////////////

////////////////////////////////////
// Receive switch5
if(varName.equals("switch5")){
    switch5 = (int)varValue;
}
////////////////////////////////////
}

public static void main(String[] args) {
    try {
        FederateConfigParser federateConfigParser =
            new FederateConfigParser();
        FederateConfig federateConfig =
            federateConfigParser.parseArgs(args, FederateConfig.class);
        Database federate =
            new Database(federateConfig);
        federate.execute();
        log.info("Done.");
        System.exit(0);
    }
}

```

```
        catch (Exception e) {  
            log.error(e);  
            System.exit(1);  
        }  
    }  
}
```

Appendix L. Occupancy Simulator and Thermostat Control

```
package org.webgme.guest.thermostat;

import org.webgme.guest.thermostat.rti.*;

import org.cpswt.config.FederateConfig;
import org.cpswt.config.FederateConfigParser;
import org.cpswt.hla.InteractionRoot;
import org.cpswt.hla.base.AdvanceTimeRequest;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.util.Random;

// Define the Thermostat type of federate for the federation.

public class Thermostat extends ThermostatBase {
    private final static Logger log = LogManager.getLogger();

    private double currentTime = 0;

    public Thermostat(FederateConfig params) throws Exception {
        super(params);
    }

    int hour=0, nexthour=0, quarter=0, fivemin=0, onemin=0, simulatetime=0;

    double r1 =0;
    double Preset_cool=23, Preset_heat=21;
    double event_p=0, duration_p=1, duration_q=0, nextevent_p=0;
    int occupancy = 2, check = 0, p=0, r2=0;
    double outTemp=23.0, coolTemp=23, heatTemp=21, zoneTemp=23.0, zoneHumidity=0,clo=1;
    String varname;
    double value;

    double Last_cool=23, Last_heat=21;
    int Fuzzycool=0,Fuzzyheat=0;
```

```

private void checkReceivedSubscriptions() {
    InteractionRoot interaction = null;
    while ((interaction = getNextInteractionNoWait()) != null) {
        if (interaction instanceof ReceiveEP) {
            handleInteractionClass((ReceiveEP) interaction);
        }
        else {
            log.debug("unhandled interaction: {}", interaction.getClassName());
        }
    }
}

```

```

private void execute() throws Exception {
    if(super.isLateJoiner()) {
        log.info("turning off time regulation (late joiner)");
        currentTime = super.getLBTS() - super.getLookAhead();
        super.disableTimeRegulation();
    }
}

```

```

////////////////////
// TODO perform basic initialization below //
////////////////////

```

```

double[] pro;

```

```

// allocating memory for 24 doubles.
pro = new double[24];

```

```

// initialize the elements of the probability array
pro[0] = 0.875;
pro[1] = 0.817;
pro[2] = 0.867;
pro[3] = 0.783;
pro[4] = 0.8;
pro[5] = 0.708;
pro[6] = 0.767;
pro[7] = 0.683;
pro[8] = 0.658;
pro[9] = 0.5;
pro[10] = 0.517;
pro[11] = 0.358;
pro[12] = 0.225;
pro[13] = 0.167;
pro[14] = 0.192;
pro[15] = 0.25;
pro[16] = 0.542;
pro[17] = 0.642;

```

```

pro[18] = 0.608;
pro[19] = 0.692;
pro[20] = 0.85;
pro[21] = 0.858;
pro[22] = 0.883;
pro[23] = 0.85;

AdvanceTimeRequest atr = new AdvanceTimeRequest(currentTime);
putAdvanceTimeRequest(atr);

if(!super.isLateJoiner()) {
    log.info("waiting on readyToPopulate...");
    readyToPopulate();
    log.info("...synchronized on readyToPopulate");
}

////////////////////////////////////
// TODO perform initialization that depends on other federates below //
////////////////////////////////////

if(!super.isLateJoiner()) {
    log.info("waiting on readyToRun...");
    readyToRun();
    log.info("...synchronized on readyToRun");
}

startAdvanceTimeThread();
log.info("started logical time progression");

while (!exitCondition) {
    atr.requestSyncStart();
    enteredTimeGrantedState();

    checkReceivedSubscriptions();

    System.out.println("Logical Time is = " + currentTime);
    // quarter = (int) (currentTime%96);
    // fivemin = (int) (currentTime%288);
    onemin = (int) (currentTime%1440);

    hour = (int) (onemin/60);
    System.out.println("Hour is = " + hour);

```

```

if (hour==simulatetime){
    System.out.println("Simulate at this time step!!!!");
    event_p=pro[hour];
    System.out.println("Occupied Probability: "+ event_p);

    // Occupancy Duration Simulator Based on Monte Carlo Method
    while (check ==0){
//        r2 = (int) (Math.random()*24+1);
        r2=1;
        System.out.println("Random Number(Duration): "+ r2 );
        for (int i = 0; i < r2; i++){
            p=hour+i;
            if (p>23){
                p=p-24;
            }
            event_p = pro[p];

            duration_p = duration_p * event_p ;

            duration_q = duration_q *(1 - event_p) ;

        }

        System.out.println("The probability of OCCUPIED for "+ r2 +" hours is "+ duration_p );

        System.out.println("The probability of NOT OCCUPIED for "+ r2 +" hours is "+ duration_q );

        r1 = (double) (Math.random());
        System.out.println("Random Number(Event): "+ r1 );
        if (r1 < duration_p){
            System.out.println("Continuously Occupied for "+r2+ " Hours Accepted!!!!" );
            check = 1;
            simulatetime = simulatetime + r2;
            occupancy =1;
            if (simulatetime>23){
                simulatetime=simulatetime-24;
            }
            System.out.println("Next simulating time at hour = "+ simulatetime );
        } else if(r1 > (1-duration_q)){
            System.out.println("Continuously Not Occupied for "+r2+ " Hours Accepted!!!!" );
            check = 1;
            simulatetime = simulatetime + r2;

```



```

        occupancy=0;
        if (simulatetime>23){
            simulatetime=simulatetime-24;
        }
        System.out.println("Next simulating time at hour = "+ simulatetime );

    } else {
        System.out.println("Event Rejected!!!" );
    }
    duration_p=1;
    duration_q=1;

}
check =0;

}else{
    System.out.println("Keep the same Occupancy information as previous timestamp.");
}

// Adaptive Control
// Always-On
if (outTemp<=10){
    heatTemp=18.9;
    coolTemp=22.9;
}else if (outTemp>=33.5){
    heatTemp=26.2;
    coolTemp=30.2;
}else {
    heatTemp = 0.31*outTemp + 17.8-2;
    coolTemp = 0.31*outTemp + 17.8+2;
}

// Fixed SetPoint

// Schedule Control

//      if (hour >8 && hour<18 ){
//          coolTemp = 32;
//          heatTemp = 12;
//      } else {
//          coolTemp = 23;
//          heatTemp = 21;
//      }

// Always-ON

```

```

coolTemp = 23;
heatTemp = 21;

// Occupancy-Driven Control

if (occupancy == 0 ){
    coolTemp = 32;
    heatTemp = 12;
}

////////////////////////////////////
// Don't Forget to delete the old occupancy and comfort file ///
////////////////////////////////////

//0.5 degree fuzzy control
double offset=0.6;

if (zoneTemp>=coolTemp+0.5-offset){
    Fuzzyheat = -1;
    Fuzzycool = -1;
}else if (zoneTemp>=coolTemp-0.5-offset){
    Fuzzyheat = -1;
}else if (zoneTemp>=heatTemp+0.5+offset){
    Fuzzyheat = -1;
    Fuzzycool = 1;
}else if (zoneTemp>=heatTemp-0.5+offset){

    Fuzzycool = 1;
}else{
    Fuzzyheat = 1;
    Fuzzycool = 1;
}
coolTemp = coolTemp -offset+ Fuzzycool*offset;
heatTemp = heatTemp +offset+ Fuzzyheat*offset;

// write to data to record occupancy information

try{
    // Create new file

    String path="/home/vagrant/EnergyPlusData/Adaptive_Test_Occupancy.txt";
    File file = new File(path);

    // If file doesn't exists, then create it

```

```

        if (!file.exists()) {
            file.createNewFile();
        }

        FileWriter fw = new FileWriter(file.getAbsolutePath(), true);
        BufferedWriter bw = new BufferedWriter(fw);

        // Write in file
        bw.write(currentTime + "\t" + occupancy + "\n");

        // Close connection
        bw.close();
    }
    catch (Exception e) {
        System.out.println(e);
    }
}
//=====

////////////////////////////////////
// TODO send interactions that must be sent every logical //
// time step below                                     //
////////////////////////////////////

    // Send the Cooling Setpoint interaction's.

    SendEP coolT = create_SendEP();
    coolT.set_varName("epGetStartCooling");
    coolT.set_value(coolTemp);
    System.out.println("Send coolTemp interaction as epGetStartCooling: " + coolTemp);
    coolT.sendInteraction(getLRC(), currentTime + getLookAhead());

    // Send the Heating Setpoint interaction's.

    SendEP heatT = create_SendEP();
    heatT.set_varName("epGetStartHeating");
    heatT.set_value(heatTemp);
    System.out.println("Send heatTemp interaction as epGetStartCooling: " + heatTemp);
    heatT.sendInteraction(getLRC(), currentTime + getLookAhead());

////////////////////////////////////
// TODO break here if ready to resign and break out of while loop //
////////////////////////////////////

if (!exitCondition) {
    currentTime += super.getStepSize();
    AdvanceTimeRequest newATR =
        new AdvanceTimeRequest(currentTime);

```

```

        putAdvanceTimeRequest(newATR);
        atr.requestSyncEnd();
        atr = newATR;
    }
}

// call exitGracefully to shut down federate
exitGracefully();

////////////////////////////////////
// TODO Perform whatever cleanups are needed before exiting the app //
////////////////////////////////////
}

private void handleInteractionClass(ReceiveEP interaction) {
    //////////////////////////////////////
    // TODO implement how to handle reception of the interaction //
    //////////////////////////////////////
    varname = interaction.get_varName();
    value = interaction.get_value();

    if(varname.equals("epSendOutdoorAirTemp")) {
        outTemp = value;
        System.out.println("Received Out Temp interaction as: " + varname + value);
    }

    if(varname.equals("epSendZoneMeanAirTemp")) {
        zoneTemp = value;
        System.out.println("Received Zone Temp interaction as: " + varname + value);
    }

    if(varname.equals("epSendZoneHumidity")) {
        zoneHumidity = value;
        System.out.println("Received Humidity interaction as: " + varname + value);
    }

    if(varname.equals("epSendClo")) {
        clo = value;
        System.out.println("Received clo interaction as: " + varname + value);
    }
}

}

public static void main(String[] args) {
    try {
        FederateConfigParser federateConfigParser =
            new FederateConfigParser();
        FederateConfig federateConfig =
            federateConfigParser.parseArgs(args, FederateConfig.class);
        Thermostat federate =
            new Thermostat(federateConfig);
    }
}

```

```
        federate.execute();
        log.info("Done.");
        System.exit(0);
    }
    catch (Exception e) {
        log.error(e);
        System.exit(1);
    }
}
}
```

Appendix M. Comfort Level Evaluate

```
package org.webgme.guest.comfortcheck;

import org.webgme.guest.comfortcheck.rti.*;

import org.cpswt.config.FederateConfig;
import org.cpswt.config.FederateConfigParser;
import org.cpswt.hla.InteractionRoot;
import org.cpswt.hla.base.AdvanceTimeRequest;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.lang.Math;

// Define the ComfortCheck type of federate for the federation.

public class ComfortCheck extends ComfortCheckBase {
    private final static Logger log = LogManager.getLogger();

    private double currentTime = 0;

    public ComfortCheck(FederateConfig params) throws Exception {
        super(params);
    }

    double outTemp=23.0, zoneTemp=23.0, zoneHumidity=0,clo=1,tempDiff=5, optset=20;
    String varname;
    double value;
    int Comfort=8;

    private void checkReceivedSubscriptions() {
        InteractionRoot interaction = null;
        while ((interaction = getNextInteractionNoWait()) != null) {
            if (interaction instanceof ReceiveEP) {
                handleInteractionClass((ReceiveEP) interaction);
            }
            else {
                log.debug("unhandled interaction: {}", interaction.getClassName());
            }
        }
    }
}
```

```

private void execute() throws Exception {
    if(super.isLateJoiner()) {
        log.info("turning off time regulation (late joiner)");
        currentTime = super.getLBTS() - super.getLookAhead();
        super.disableTimeRegulation();
    }

    //////////////////////////////////////
    // TODO perform basic initialization below //
    //////////////////////////////////////

    AdvanceTimeRequest atr = new AdvanceTimeRequest(currentTime);
    putAdvanceTimeRequest(atr);

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToPopulate...");
        readyToPopulate();
        log.info("...synchronized on readyToPopulate");
    }

    //////////////////////////////////////
    // TODO perform initialization that depends on other federates below //
    //////////////////////////////////////

    if(!super.isLateJoiner()) {
        log.info("waiting on readyToRun...");
        readyToRun();
        log.info("...synchronized on readyToRun");
    }

    startAdvanceTimeThread();
    log.info("started logical time progression");

    while (!exitCondition) {
        atr.requestSyncStart();
        enteredTimeGrantedState();

        checkReceivedSubscriptions();

        if (outTemp<10){
            optset = 20.9;
        }else if(outTemp>33.5){
            optset = 28.2;
        }else {
            optset = 0.31*outTemp+17.8;
        }

        tempDiff = Math.abs(zoneTemp-optset);
    }

```

```

if (tempDiff<2){
    Comfort = 0;
}else if (tempDiff<3){
    Comfort = 1;
}else{
    Comfort = 2;
}

// write to data to record comfort information

try{
    // Create new file

    String path="/home/vagrant/EnergyPlusData/Adaptive_Test_Comfort.txt";
    File file = new File(path);

    // If file doesn't exists, then create it
    if (!file.exists()) {
        file.createNewFile();
    }

    FileWriter fw = new FileWriter(file.getAbsolutePath(),true);
    BufferedWriter bw = new BufferedWriter(fw);

    // Write in file
    bw.write(currentTime + "\t" + Comfort + "\n" );

    // Close connection
    bw.close();
}
catch(Exception e){
    System.out.println(e);
}
}

//=====

////////////////////////////////////
// TODO break here if ready to resign and break out of while loop //
////////////////////////////////////

if (!exitCondition) {
    currentTime += super.getStepSize();
    AdvanceTimeRequest newATR =
        new AdvanceTimeRequest(currentTime);
    putAdvanceTimeRequest(newATR);
    atr.requestSyncEnd();
    atr = newATR;
}
}

```



```

// call exitGracefully to shut down federate
exitGracefully();

////////////////////////////////////
// TODO Perform whatever cleanups are needed before exiting the app //
////////////////////////////////////
}

private void handleInteractionClass(ReceiveEP interaction) {
    //////////////////////////////////
    // TODO implement how to handle reception of the interaction //
    //////////////////////////////////
    varname = interaction.get_varName();
    value = interaction.get_value();

    if(varname.equals("epSendOutdoorAirTemp")) {
        outTemp = value;
        System.out.println("Received Out Temp interaction as: " + varname + value);
    }

    if(varname.equals("epSendZoneMeanAirTemp")) {
        zoneTemp = value;
        System.out.println("Received Zone Temp interaction as: " + varname + value);
    }

    if(varname.equals("epSendZoneHumidity")) {
        zoneHumidity = value;
        System.out.println("Received Humidity interaction as: " + varname + value);
    }

    if(varname.equals("epSendClo")) {
        clo = value;
        System.out.println("Received clo interaction as: " + varname + value);
    }
}

public static void main(String[] args) {
    try {
        FederateConfigParser federateConfigParser =
            new FederateConfigParser();
        FederateConfig federateConfig =
            federateConfigParser.parseArgs(args, FederateConfig.class);
        ComfortCheck federate =
            new ComfortCheck(federateConfig);
        federate.execute();
        log.info("Done.");
        System.exit(0);
    }
    catch (Exception e) {
        log.error(e);
        System.exit(1);
    }
}

```

}
}
}

Appendix N. EnergyPlus IDF Incorporated IDF Model

!-Generator IDFEditor 1.49

!-Option SortedOrder ViewInIPunits

!-NOTE: All comments with '!' are ignored by the IDFEditor and are generated automatically.

!- Use '!' comments if they need to be retained when using the IDFEditor.

!- ===== ALL OBJECTS IN CLASS: VERSION =====

! Varlist contains a list of variables used

! Simple glazing system considered

Version,

8.7; !- Version Identifier

!- ===== ALL OBJECTS IN CLASS: SIMULATIONCONTROL =====

SimulationControl,

Yes, !- Do Zone Sizing Calculation

Yes, !- Do System Sizing Calculation

No, !- Do Plant Sizing Calculation

Yes, !- Run Simulation for Sizing Periods

Yes; !- Run Simulation for Weather File Run Periods

!- ===== ALL OBJECTS IN CLASS: BUILDING =====

Building,

SF_Florida_Miami.Intl.AP.722020_gasfurnace_crawlspace_IECC_2012, !- Name

0, !- North Axis {deg}

Suburbs, !- Terrain

0.04, !- Loads Convergence Tolerance Value

0.4, !- Temperature Convergence Tolerance Value {deltaC}

FullExterior, !- Solar Distribution

25, !- Maximum Number of Warmup Days

6; !- Minimum Number of Warmup Days

!- ===== ALL OBJECTS IN CLASS: SURFACECONVECTIONALGORITHM:INSIDE =====

SurfaceConvectionAlgorithm:Inside,

```

TARP;          !- Algorithm

!- ===== ALL OBJECTS IN CLASS: TIMESTEP =====

Timestep,
  60;          !- Number of Timesteps per Hour

!- ===== ALL OBJECTS IN CLASS: SITE:LOCATION =====

! Location and design-day objects created by:
! Site:Location and design-day objects created by:
!
! /tools/bin/ddy2idf
/phome/weather/EnergyPlus/tmy3/usa/zip/USA_FL_Miami.Intl.AP.722020_TMY3.ddy
!
Site:Location,
  Miami Intl Ap_FL_USA Design_Conditions, !- Name
  25.82,          !- Latitude {deg}
  -80.30,         !- Longitude {deg}
  -5.00,         !- Time Zone {hr}
  11.00;         !- Elevation {m}

!- ===== ALL OBJECTS IN CLASS: SIZINGPERIOD:DESIGNDAY =====

SizingPeriod:DesignDay,
  Miami Intl Ap Ann Htg 99.6% Condns DB, !- Name
  1,             !- Month
  21,            !- Day of Month
  WinterDesignDay, !- Day Type
  8.7,           !- Maximum Dry-Bulb Temperature {C}
  0.0,           !- Daily Dry-Bulb Temperature Range {deltaC}
  ,              !- Dry-Bulb Temperature Range Modifier Type
  ,              !- Dry-Bulb Temperature Range Modifier Day Schedule Name
  Wetbulb,       !- Humidity Condition Type
  8.7,           !- Wetbulb or DewPoint at Maximum Dry-Bulb {C}
  ,              !- Humidity Condition Day Schedule Name
  ,              !- Humidity Ratio at Maximum Dry-Bulb {kgWater/kgDryAir}
  ,              !- Enthalpy at Maximum Dry-Bulb {J/kg}
  ,              !- Daily Wet-Bulb Temperature Range {deltaC}
  101193.,       !- Barometric Pressure {Pa}
  3.8,           !- Wind Speed {m/s}
  340,           !- Wind Direction {deg}
  No,            !- Rain Indicator
  No,            !- Snow Indicator
  No,            !- Daylight Saving Time Indicator
  ASHRAEClearSky, !- Solar Model Indicator
  ,              !- Beam Solar Day Schedule Name

```

, !- Diffuse Solar Day Schedule Name
 , !- ASHRAE Clear Sky Optical Depth for Beam Irradiance (taub) {dimensionless}
 , !- ASHRAE Clear Sky Optical Depth for Diffuse Irradiance (taud) {dimensionless}
 0.00; !- Sky Clearness

SizingPeriod:DesignDay,

 Miami Intl Ap Ann Clg .4% Condns DB=>MWB, !- Name
 7, !- Month
 21, !- Day of Month
 SummerDesignDay, !- Day Type
 33.2, !- Maximum Dry-Bulb Temperature {C}
 6.7, !- Daily Dry-Bulb Temperature Range {deltaC}
 , !- Dry-Bulb Temperature Range Modifier Type
 , !- Dry-Bulb Temperature Range Modifier Day Schedule Name
 Wetbulb, !- Humidity Condition Type
 25.3, !- Wetbulb or DewPoint at Maximum Dry-Bulb {C}
 , !- Humidity Condition Day Schedule Name
 , !- Humidity Ratio at Maximum Dry-Bulb {kgWater/kgDryAir}
 , !- Enthalpy at Maximum Dry-Bulb {J/kg}
 , !- Daily Wet-Bulb Temperature Range {deltaC}
 101193., !- Barometric Pressure {Pa}
 4.5, !- Wind Speed {m/s}
 140, !- Wind Direction {deg}
 No, !- Rain Indicator
 No, !- Snow Indicator
 No, !- Daylight Saving Time Indicator
 ASHRAEClearSky, !- Solar Model Indicator
 , !- Beam Solar Day Schedule Name
 , !- Diffuse Solar Day Schedule Name
 , !- ASHRAE Clear Sky Optical Depth for Beam Irradiance (taub) {dimensionless}
 , !- ASHRAE Clear Sky Optical Depth for Diffuse Irradiance (taud) {dimensionless}
 1.00; !- Sky Clearness

!- ===== ALL OBJECTS IN CLASS: RUNPERIOD =====

RunPeriod,

 annual, !- Name
 1, !- Begin Month
 1, !- Begin Day of Month
 12, !- End Month
 31, !- End Day of Month
 UseWeatherFile, !- Day of Week for Start Day
 Yes, !- Use Weather File Holidays and Special Days
 Yes, !- Use Weather File Daylight Saving Period
 No, !- Apply Weekend Holiday Rule
 Yes, !- Use Weather File Rain Indicators
 Yes, !- Use Weather File Snow Indicators
 1; !- Number of Times Runperiod to be Repeated

!- ===== ALL OBJECTS IN CLASS: SITE:WATERMAINSTEMPERATURE =====

```
! Weather_file = USA_FL_Miami.Intl.AP.722020_TMY3.epw;
! Using the stat2idf and the ddy2idf script...
! Water mains temperature correlation object created by:
! Water mains temperature object created by:
!
! /phome/resstd/vm/bin/stat2idf
/phome/weather/EnergyPlus/tmy3/usa/zip/USA_FL_Miami.Intl.AP.722020_TMY3.stat
!
Site:WaterMainsTemperature,
  Correlation,          !- Calculation Method
  ,                    !- Temperature Schedule Name
  24.4916666666667,     !- Annual Average Outdoor Air Temperature {C}
  8.7;                 !- Maximum Difference In Monthly Average Outdoor Air Temperatures {deltaC}
```

!- ===== ALL OBJECTS IN CLASS: SCHEDULETYPELIMITS =====

```
ScheduleTypeLimits,
  any number;          !- Name
```

```
ScheduleTypeLimits,
  On/Off,              !- Name
  0,                   !- Lower Limit Value
  1,                   !- Upper Limit Value
  Discrete;           !- Numeric Type
```

```
ScheduleTypeLimits,
  control_type,        !- Name
  0,                   !- Lower Limit Value
  4,                   !- Upper Limit Value
  Discrete;           !- Numeric Type
```

```
ScheduleTypeLimits,
  fraction,            !- Name
  0,                   !- Lower Limit Value
  1,                   !- Upper Limit Value
  Continuous;         !- Numeric Type
```

```
ScheduleTypeLimits,
  Temperature,         !- Name
  -60,                 !- Lower Limit Value
  200,                 !- Upper Limit Value
  Continuous;         !- Numeric Type
```

```
ScheduleTypeLimits,
  myTemps,             !- Name
```

, !- Lower Limit Value
 , !- Upper Limit Value
 Continuous, !- Numeric Type
 Temperature; !- Unit Type

!- ===== ALL OBJECTS IN CLASS: SCHEDULE:DAY:HOURLY =====

Schedule:Day:Hourly,
 S-Lighting-day, !- Name
 Fraction, !- Schedule Type Limits Name
 0.16, !- Hour 1
 0.15, !- Hour 2
 0.16, !- Hour 3
 0.18, !- Hour 4
 0.23, !- Hour 5
 0.45, !- Hour 6
 0.4, !- Hour 7
 0.26, !- Hour 8
 0.19, !- Hour 9
 0.16, !- Hour 10
 0.12, !- Hour 11
 0.11, !- Hour 12
 0.16, !- Hour 13
 0.17, !- Hour 14
 0.25, !- Hour 15
 0.27, !- Hour 16
 0.34, !- Hour 17
 0.55, !- Hour 18
 0.55, !- Hour 19
 0.88, !- Hour 20
 1, !- Hour 21
 0.86, !- Hour 22
 0.51, !- Hour 23
 0.28; !- Hour 24

!
 ! *** Occupancy and Lighting factors taken from NREL's benchmark model ***
 !

Schedule:Day:Hourly,
 OccupancyDay, !- Name
 Fraction, !- Schedule Type Limits Name
 1.00000, !- Hour 1
 1.00000, !- Hour 2
 1.00000, !- Hour 3
 1.00000, !- Hour 4
 1.00000, !- Hour 5
 1.00000, !- Hour 6
 1.00000, !- Hour 7

0.88310,	!- Hour 8
0.40861,	!- Hour 9
0.24189,	!- Hour 10
0.24189,	!- Hour 11
0.24189,	!- Hour 12
0.24189,	!- Hour 13
0.24189,	!- Hour 14
0.24189,	!- Hour 15
0.24189,	!- Hour 16
0.29498,	!- Hour 17
0.55310,	!- Hour 18
0.89693,	!- Hour 19
0.89693,	!- Hour 20
0.89693,	!- Hour 21
1.00000,	!- Hour 22
1.00000,	!- Hour 23
1.00000;	!- Hour 24

Schedule:Day:Hourly,

LightingProfileDay,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0625,	!- Hour 1
0.0625,	!- Hour 2
0.0625,	!- Hour 3
0.0625,	!- Hour 4
0.1875,	!- Hour 5
0.390625,	!- Hour 6
0.4375,	!- Hour 7
0.390625,	!- Hour 8
0.171875,	!- Hour 9
0.1171875,	!- Hour 10
0.1171875,	!- Hour 11
0.1171875,	!- Hour 12
0.1171875,	!- Hour 13
0.1171875,	!- Hour 14
0.1171875,	!- Hour 15
0.203125,	!- Hour 16
0.4375,	!- Hour 17
0.609375,	!- Hour 18
0.8203125,	!- Hour 19
0.984375,	!- Hour 20
1,	!- Hour 21
0.6875,	!- Hour 22
0.3828125,	!- Hour 23
0.15625;	!- Hour 24

Schedule:Day:Hourly,

ExteriorLightingProfileDay,	!- Name
Fraction,	!- Schedule Type Limits Name

1,	!- Hour 1
1,	!- Hour 2
1,	!- Hour 3
1,	!- Hour 4
1,	!- Hour 5
1,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14
0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
1,	!- Hour 19
1,	!- Hour 20
1,	!- Hour 21
1,	!- Hour 22
1,	!- Hour 23
1;	!- Hour 24

Schedule:Day:Hourly,

LightingProfileDay_EELighting, !- Name	
Fraction,	!- Schedule Type Limits Name
0.06875,	!- Hour 1
0.06875,	!- Hour 2
0.06875,	!- Hour 3
0.06875,	!- Hour 4
0.20625,	!- Hour 5
0.4296875,	!- Hour 6
0.48125,	!- Hour 7
0.4296875,	!- Hour 8
0.1890625,	!- Hour 9
0.12890625,	!- Hour 10
0.12890625,	!- Hour 11
0.12890625,	!- Hour 12
0.12890625,	!- Hour 13
0.12890625,	!- Hour 14
0.12890625,	!- Hour 15
0.2234375,	!- Hour 16
0.48125,	!- Hour 17
0.6703125,	!- Hour 18
0.90234375,	!- Hour 19
1,	!- Hour 20
1,	!- Hour 21

0.75625, !- Hour 22
0.42109375, !- Hour 23
0.171875; !- Hour 24

Schedule:Day:Hourly,

RefrigeratorDay, !- Name
Fraction, !- Schedule Type Limits Name
0.8, !- Hour 1
0.782696177062374, !- Hour 2
0.765593561368209, !- Hour 3
0.742857142857143, !- Hour 4
0.731388329979879, !- Hour 5
0.731388329979879, !- Hour 6
0.759959758551308, !- Hour 7
0.8, !- Hour 8
0.817102615694165, !- Hour 9
0.828571428571429, !- Hour 10
0.8, !- Hour 11
0.8, !- Hour 12
0.839839034205231, !- Hour 13
0.839839034205231, !- Hour 14
0.828571428571429, !- Hour 15
0.839839034205231, !- Hour 16
0.885714285714286, !- Hour 17
0.971428571428572, !- Hour 18
1, !- Hour 19
0.971428571428572, !- Hour 20
0.942857142857143, !- Hour 21
0.925553319919517, !- Hour 22
0.885714285714286, !- Hour 23
0.828571428571429; !- Hour 24

Schedule:Day:Hourly,

MiscPlugLoadDay, !- Name
Fraction, !- Schedule Type Limits Name
0.607490272373541, !- Hour 1
0.559338521400778, !- Hour 2
0.552853437094682, !- Hour 3
0.545071335927367, !- Hour 4
0.524481193255512, !- Hour 5
0.585278858625162, !- Hour 6
0.676232166018158, !- Hour 7
0.718547341115435, !- Hour 8
0.607490272373541, !- Hour 9
0.517023346303502, !- Hour 10
0.529182879377432, !- Hour 11
0.529345006485084, !- Hour 12
0.520428015564202, !- Hour 13
0.538424124513619, !- Hour 14

0.568741893644617,	!- Hour 15
0.600356679636835,	!- Hour 16
0.71011673151751,	!- Hour 17
0.862678339818418,	!- Hour 18
0.936608300907912,	!- Hour 19
0.966763942931258,	!- Hour 20
1,	!- Hour 21
0.976653696498055,	!- Hour 22
0.845168612191959,	!- Hour 23
0.73443579766537;	!- Hour 24

Schedule:Day:Hourly,

CookingRangeDay,	!- Name
Fraction,	!- Schedule Type Limits Name
0.04715848452508,	!- Hour 1
0.04715848452508,	!- Hour 2
0.0235458911419424,	!- Hour 3
0.0235458911419424,	!- Hour 4
0.04715848452508,	!- Hour 5
0.0707043756670224,	!- Hour 6
0.165088046958378,	!- Hour 7
0.283017609391676,	!- Hour 8
0.306563500533618,	!- Hour 9
0.320771077908218,	!- Hour 10
0.283017609391676,	!- Hour 11
0.330176093916756,	!- Hour 12
0.377334578441836,	!- Hour 13
0.306563500533618,	!- Hour 14
0.292422625400213,	!- Hour 15
0.377334578441836,	!- Hour 16
0.613193703308431,	!- Hour 17
1,	!- Hour 18
0.778348452508004,	!- Hour 19
0.400947171824973,	!- Hour 20
0.235859124866596,	!- Hour 21
0.165088046958378,	!- Hour 22
0.103721985058698,	!- Hour 23
0.0707043756670224;	!- Hour 24

Schedule:Day:Hourly,

DishwasherWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name
0.129162158913331,	!- Hour 1
0.0573867051328282,	!- Hour 2
0.0429979564851542,	!- Hour 3
0.0286933525664145,	!- Hour 4
0.0286933525664145,	!- Hour 5
0.0860800576992427,	!- Hour 6
0.172244260127419,	!- Hour 7

0.258408462555596,	!- Hour 8
0.488207717273711,	!- Hour 9
0.545594422406539,	!- Hour 10
0.473818968626037,	!- Hour 11
0.402043514845535,	!- Hour 12
0.344572664983772,	!- Hour 13
0.387654766197861,	!- Hour 14
0.315879312417358,	!- Hour 15
0.301490563769684,	!- Hour 16
0.315879312417358,	!- Hour 17
0.416348118764274,	!- Hour 18
0.732311575910566,	!- Hour 19
0.933333333333333,	!- Hour 20
0.761004928476981,	!- Hour 21
0.559983171054213,	!- Hour 22
0.373266017550187,	!- Hour 23
0.258408462555596;	!- Hour 24

Schedule:Day:Hourly,

DishwasherWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.13838802740714,	!- Hour 1
0.0614857554994591,	!- Hour 2
0.0460692390912369,	!- Hour 3
0.0307428777497295,	!- Hour 4
0.0307428777497295,	!- Hour 5
0.0922286332491886,	!- Hour 6
0.184547421565092,	!- Hour 7
0.276866209880995,	!- Hour 8
0.523079697078976,	!- Hour 9
0.584565452578435,	!- Hour 10
0.507663180670754,	!- Hour 11
0.430760908763072,	!- Hour 12
0.369184998196899,	!- Hour 13
0.41534439235485,	!- Hour 14
0.338442120447169,	!- Hour 15
0.323025604038947,	!- Hour 16
0.338442120447169,	!- Hour 17
0.44608727010458,	!- Hour 18
0.784619545618464,	!- Hour 19
1,	!- Hour 20
0.815362423368193,	!- Hour 21
0.599981968986657,	!- Hour 22
0.399927875946628,	!- Hour 23
0.276866209880995;	!- Hour 24

Schedule:Day:Hourly,

DishwasherVacation,	!- Name
Fraction,	!- Schedule Type Limits Name

0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14
0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

ClothesWasherWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0887931470412736,	!- Hour 1
0.0710155041111689,	!- Hour 2
0.0354602182509584,	!- Hour 3
0.0354602182509584,	!- Hour 4
0.0710155041111689,	!- Hour 5
0.106570789971378,	!- Hour 6
0.213141579942758,	!- Hour 7
0.461933513354977,	!- Hour 8
0.69294780383709,	!- Hour 9
0.817391304347826,	!- Hour 10
0.799613661417722,	!- Hour 11
0.710725446767196,	!- Hour 12
0.639614875046775,	!- Hour 13
0.568599370935607,	!- Hour 14
0.497488799215186,	!- Hour 15
0.461933513354977,	!- Hour 16
0.479711156285081,	!- Hour 17
0.461933513354977,	!- Hour 18
0.461933513354977,	!- Hour 19
0.461933513354977,	!- Hour 20
0.461933513354977,	!- Hour 21

0.444155870424871,	!- Hour 22
0.302029794593282,	!- Hour 23
0.159903718761693;	!- Hour 24

Schedule:Day:Hourly,

ClothesWasherWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.108629913933473,	!- Hour 1
0.086880669923238,	!- Hour 2
0.0433821819027681,	!- Hour 3
0.0433821819027681,	!- Hour 4
0.086880669923238,	!- Hour 5
0.130379157943708,	!- Hour 6
0.260758315887416,	!- Hour 7
0.565131425913003,	!- Hour 8
0.847755291928355,	!- Hour 9
1,	!- Hour 10
0.978250755989765,	!- Hour 11
0.86950453593859,	!- Hour 12
0.782507559897651,	!- Hour 13
0.695626889974413,	!- Hour 14
0.608629913933473,	!- Hour 15
0.565131425913003,	!- Hour 16
0.586880669923238,	!- Hour 17
0.565131425913003,	!- Hour 18
0.565131425913003,	!- Hour 19
0.565131425913003,	!- Hour 20
0.565131425913003,	!- Hour 21
0.543382181902768,	!- Hour 22
0.36950453593859,	!- Hour 23
0.195626889974413;	!- Hour 24

Schedule:Day:Hourly,

ClothesWasherVacation,	!- Name
Fraction,	!- Schedule Type Limits Name
0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14

0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

dhw_profile_day,	!- Name
fraction,	!- Schedule Type Limits Name
0.006,	!- Hour 1
0.003,	!- Hour 2
0.001,	!- Hour 3
0.001,	!- Hour 4
0.003,	!- Hour 5
0.021,	!- Hour 6
0.075,	!- Hour 7
0.079,	!- Hour 8
0.076,	!- Hour 9
0.067,	!- Hour 10
0.061,	!- Hour 11
0.05,	!- Hour 12
0.042,	!- Hour 13
0.038,	!- Hour 14
0.033,	!- Hour 15
0.038,	!- Hour 16
0.043,	!- Hour 17
0.058,	!- Hour 18
0.068,	!- Hour 19
0.065,	!- Hour 20
0.06,	!- Hour 21
0.047,	!- Hour 22
0.041,	!- Hour 23
0.024;	!- Hour 24

Schedule:Day:Hourly,

ClothesDryerWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0996818663838815,	!- Hour 1
0.0598091198303289,	!- Hour 2
0.0398727465535526,	!- Hour 3
0.0199363732767763,	!- Hour 4
0.0398727465535526,	!- Hour 5
0.0598091198303289,	!- Hour 6
0.15949098621421,	!- Hour 7

0.31898197242842,	!- Hour 8
0.486427370202556,	!- Hour 9
0.685791102970318,	!- Hour 10
0.785472969354199,	!- Hour 11
0.817391304347826,	!- Hour 12
0.745600222800647,	!- Hour 13
0.677836691410393,	!- Hour 14
0.610073160020138,	!- Hour 15
0.578154825026511,	!- Hour 16
0.558218451749735,	!- Hour 17
0.546236490032885,	!- Hour 18
0.518345705196183,	!- Hour 19
0.510391293636256,	!- Hour 20
0.526300116756109,	!- Hour 21
0.546236490032885,	!- Hour 22
0.438600212089077,	!- Hour 23
0.239236479321316;	!- Hour 24

Schedule:Day:Hourly,

ClothesDryerWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.121951219512195,	!- Hour 1
0.0731707317073171,	!- Hour 2
0.0487804878048781,	!- Hour 3
0.024390243902439,	!- Hour 4
0.0487804878048781,	!- Hour 5
0.0731707317073171,	!- Hour 6
0.195121951219512,	!- Hour 7
0.390243902439024,	!- Hour 8
0.59509731460951,	!- Hour 9
0.8389997536339,	!- Hour 10
0.960950973146095,	!- Hour 11
1,	!- Hour 12
0.912170485341217,	!- Hour 13
0.829268292682927,	!- Hour 14
0.746366100024637,	!- Hour 15
0.707317073170732,	!- Hour 16
0.682926829268293,	!- Hour 17
0.668268046316827,	!- Hour 18
0.634146341463415,	!- Hour 19
0.624414880512441,	!- Hour 20
0.643877802414388,	!- Hour 21
0.668268046316827,	!- Hour 22
0.536585365853659,	!- Hour 23
0.292682926829268;	!- Hour 24

Schedule:Day:Hourly,

ClothesDryerVacation,	!- Name
Fraction,	!- Schedule Type Limits Name

0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14
0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

SinksWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name
0.178431090172197,	!- Hour 1
0.0855612360359615,	!- Hour 2
0.0641394243627467,	!- Hour 3
0.0641394243627467,	!- Hour 4
0.0855612360359615,	!- Hour 5
0.228457320962002,	!- Hour 6
0.535419281173493,	!- Hour 7
0.785298413808711,	!- Hour 8
0.828142037155143,	!- Hour 9
0.778115806365338,	!- Hour 10
0.685371962886004,	!- Hour 11
0.628163124652828,	!- Hour 12
0.613923920422985,	!- Hour 13
0.571080297076553,	!- Hour 14
0.521054066286749,	!- Hour 15
0.542475877959965,	!- Hour 16
0.606741312979612,	!- Hour 17
0.82095942971177,	!- Hour 18
0.942307692307692,	!- Hour 19
0.870985660501574,	!- Hour 20
0.713850371345691,	!- Hour 21

0.606741312979612,	!- Hour 22
0.499758265270435,	!- Hour 23
0.342622976114552;	!- Hour 24

Schedule:Day:Hourly,

SinksWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.18935544263172,	!- Hour 1
0.0907996790585718,	!- Hour 2
0.0680663278951591,	!- Hour 3
0.0680663278951591,	!- Hour 4
0.0907996790585718,	!- Hour 5
0.242444503878042,	!- Hour 6
0.568200053490238,	!- Hour 7
0.833377908531693,	!- Hour 8
0.878844610858518,	!- Hour 9
0.825755549612196,	!- Hour 10
0.727333511634127,	!- Hour 11
0.666622091468307,	!- Hour 12
0.651511099224392,	!- Hour 13
0.606044396897566,	!- Hour 14
0.552955335651244,	!- Hour 15
0.575688686814656,	!- Hour 16
0.643888740304894,	!- Hour 17
0.871222251939021,	!- Hour 18
1,	!- Hour 19
0.924311313185344,	!- Hour 20
0.757555496121958,	!- Hour 21
0.643888740304894,	!- Hour 22
0.53035571008291,	!- Hour 23
0.363599893019524;	!- Hour 24

Schedule:Day:Hourly,

SinksVacation,	!- Name
Fraction,	!- Schedule Type Limits Name
0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14

0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

ShowersWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0847763225038272,	!- Hour 1
0.0384986108748656,	!- Hour 2
0.0269887169019672,	!- Hour 3
0.0384986108748656,	!- Hour 4
0.10795486760787,	!- Hour 5
0.408799682485684,	!- Hour 6
0.933333333333333,	!- Hour 7
0.92563361115836,	!- Hour 8
0.752112037194534,	!- Hour 9
0.590100357203606,	!- Hour 10
0.47436638884164,	!- Hour 11
0.374111243408743,	!- Hour 12
0.269966547598798,	!- Hour 13
0.231388558144809,	!- Hour 14
0.200510290865794,	!- Hour 15
0.208210013040766,	!- Hour 16
0.239088280319782,	!- Hour 17
0.308544537052787,	!- Hour 18
0.335533253954754,	!- Hour 19
0.335533253954754,	!- Hour 20
0.331643703577706,	!- Hour 21
0.323943981402733,	!- Hour 22
0.231388558144809,	!- Hour 23
0.165821851788853;	!- Hour 24

Schedule:Day:Hourly,

ShowersWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0908317741112434,	!- Hour 1
0.0412485116516414,	!- Hour 2
0.0289164823949651,	!- Hour 3
0.0412485116516414,	!- Hour 4
0.115665929579861,	!- Hour 5
0.437999659806089,	!- Hour 6
1,	!- Hour 7

0.991750297669672,	!- Hour 8
0.805834325565572,	!- Hour 9
0.632250382718149,	!- Hour 10
0.508249702330328,	!- Hour 11
0.400833475080796,	!- Hour 12
0.289249872427284,	!- Hour 13
0.24791631229801,	!- Hour 14
0.214832454499064,	!- Hour 15
0.223082156829393,	!- Hour 16
0.256166014628338,	!- Hour 17
0.330583432556557,	!- Hour 18
0.359499914951522,	!- Hour 19
0.359499914951522,	!- Hour 20
0.355332539547542,	!- Hour 21
0.347082837217214,	!- Hour 22
0.24791631229801,	!- Hour 23
0.177666269773771;	!- Hour 24

Schedule:Day:Hourly,

ShowersVacation,	!- Name
Fraction,	!- Schedule Type Limits Name
0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14
0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

BathsWeekday,	!- Name
Fraction,	!- Schedule Type Limits Name

0.0549341075342269,	!- Hour 1
0.027467053767114,	!- Hour 2
0.027467053767114,	!- Hour 3
0.027467053767114,	!- Hour 4
0.0549341075342269,	!- Hour 5
0.137335268835568,	!- Hour 6
0.329675803375743,	!- Hour 7
0.412076964677084,	!- Hour 8
0.467011072211311,	!- Hour 9
0.412076964677084,	!- Hour 10
0.329675803375743,	!- Hour 11
0.247203483904022,	!- Hour 12
0.219736430136909,	!- Hour 13
0.164802322602681,	!- Hour 14
0.164802322602681,	!- Hour 15
0.164802322602681,	!- Hour 16
0.274741695841516,	!- Hour 17
0.329675803375743,	!- Hour 18
0.549483391683034,	!- Hour 19
0.714285714285714,	!- Hour 20
0.714285714285714,	!- Hour 21
0.549483391683034,	!- Hour 22
0.467011072211311,	!- Hour 23
0.274741695841516;	!- Hour 24

Schedule:Day:Hourly,

BathsWeekend,	!- Name
Fraction,	!- Schedule Type Limits Name
0.0769077505479179,	!- Hour 1
0.038453875273959,	!- Hour 2
0.038453875273959,	!- Hour 3
0.038453875273959,	!- Hour 4
0.0769077505479179,	!- Hour 5
0.192269376369795,	!- Hour 6
0.461546124726041,	!- Hour 7
0.576907750547918,	!- Hour 8
0.653815501095836,	!- Hour 9
0.576907750547918,	!- Hour 10
0.461546124726041,	!- Hour 11
0.346084877465631,	!- Hour 12
0.307631002191672,	!- Hour 13
0.230723251643754,	!- Hour 14
0.230723251643754,	!- Hour 15
0.230723251643754,	!- Hour 16
0.384638374178123,	!- Hour 17
0.461546124726041,	!- Hour 18
0.769276748356246,	!- Hour 19
1,	!- Hour 20
1,	!- Hour 21

0.769276748356246, !- Hour 22
0.653815501095836, !- Hour 23
0.384638374178123; !- Hour 24

Schedule:Day:Hourly,

BathsVacation,	!- Name
Fraction,	!- Schedule Type Limits Name
0,	!- Hour 1
0,	!- Hour 2
0,	!- Hour 3
0,	!- Hour 4
0,	!- Hour 5
0,	!- Hour 6
0,	!- Hour 7
0,	!- Hour 8
0,	!- Hour 9
0,	!- Hour 10
0,	!- Hour 11
0,	!- Hour 12
0,	!- Hour 13
0,	!- Hour 14
0,	!- Hour 15
0,	!- Hour 16
0,	!- Hour 17
0,	!- Hour 18
0,	!- Hour 19
0,	!- Hour 20
0,	!- Hour 21
0,	!- Hour 22
0,	!- Hour 23
0;	!- Hour 24

Schedule:Day:Hourly,

DHWDistDay,	!- Name
Fraction,	!- Schedule Type Limits Name
0.142553149370226,	!- Hour 1
0.0764866759550322,	!- Hour 2
0.0516611840277747,	!- Hour 3
0.0584039294664903,	!- Hour 4
0.121469841058603,	!- Hour 5
0.366180268451559,	!- Hour 6
0.833258955739606,	!- Hour 7
0.999999987228391,	!- Hour 8
0.995483090581232,	!- Hour 9
0.916474762598139,	!- Hour 10
0.800898393293271,	!- Hour 11
0.682564390940485,	!- Hour 12
0.590066600109555,	!- Hour 13
0.522371887032053,	!- Hour 14

0.466005974303267, !- Hour 15
 0.46573704253214, !- Hour 16
 0.527514234916123, !- Hour 17
 0.637905701335668, !- Hour 18
 0.727588642310272, !- Hour 19
 0.732551127624518, !- Hour 20
 0.681468471942116, !- Hour 21
 0.610804704600985, !- Hour 22
 0.464292318119835, !- Hour 23
 0.299867724445383; !- Hour 24

!- ===== ALL OBJECTS IN CLASS: SCHEDULE:WEEK:COMPACT =====

Schedule:Week:Compact,

S-Lighting-week, !- Name
 AllDays, !- DayType List 1
 S-Lighting-day; !- Schedule:Day Name 1

Schedule:Week:Compact,

RefrigeratorWeek, !- Name
 For: AllDays, !- DayType List 1
 RefrigeratorDay; !- Schedule:Day Name 1

Schedule:Week:Compact,

MiscPlugLoadWeek, !- Name
 For: AllDays, !- DayType List 1
 MiscPlugLoadDay; !- Schedule:Day Name 1

Schedule:Week:Compact,

CookingRangeWeek, !- Name
 For: AllDays, !- DayType List 1
 CookingRangeDay; !- Schedule:Day Name 1

Schedule:Week:Compact,

DishwasherWeek, !- Name
 For: Weekdays, !- DayType List 1
 DishwasherWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 DishwasherVacation, !- Schedule:Day Name 2
 For: AllOtherDays, !- DayType List 3
 DishwasherWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,

ClothesWasherWeek, !- Name
 For: Weekdays, !- DayType List 1
 ClothesWasherWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 ClothesWasherVacation, !- Schedule:Day Name 2

For: AllOtherDays, !- DayType List 3
ClothesWasherWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,
 dhw_profile_week, !- Name
 AllDays, !- DayType List 1
 dhw_profile_day; !- Schedule:Day Name 1

Schedule:Week:Compact,
 ClothesDryerWeek, !- Name
 For: Weekdays, !- DayType List 1
 ClothesDryerWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 ClothesDryerVacation, !- Schedule:Day Name 2
 For: AllOtherDays, !- DayType List 3
 ClothesDryerWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,
 SinksWeek, !- Name
 For: Weekdays, !- DayType List 1
 SinksWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 SinksVacation, !- Schedule:Day Name 2
 For: AllOtherDays, !- DayType List 3
 SinksWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,
 ShowersWeek, !- Name
 For: Weekdays, !- DayType List 1
 ShowersWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 ShowersVacation, !- Schedule:Day Name 2
 For: AllOtherDays, !- DayType List 3
 ShowersWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,
 BathsWeek, !- Name
 For: Weekdays, !- DayType List 1
 BathsWeekday, !- Schedule:Day Name 1
 For: CustomDay1, !- DayType List 2
 BathsVacation, !- Schedule:Day Name 2
 For: AllOtherDays, !- DayType List 3
 BathsWeekend; !- Schedule:Day Name 3

Schedule:Week:Compact,
 DHWDistWeek, !- Name
 For: AllDays, !- DayType List 1
 DHWDistDay; !- Schedule:Day Name 1

Schedule:Week:Compact,
 OccupancyWeek, !- Name
 AllDays, !- DayType List 1
 OccupancyDay; !- Schedule:Day Name 1

Schedule:Week:Compact,
 LightingProfileWeek, !- Name
 For: AllDays, !- DayType List 1
 LightingProfileDay; !- Schedule:Day Name 1

Schedule:Week:Compact,
 LightingProfileWeek_EELighting, !- Name
 For: AllDays, !- DayType List 1
 LightingProfileDay_EELighting; !- Schedule:Day Name 1

Schedule:Week:Compact,
 ExteriorLightingProfileWeek, !- Name
 For: AllDays, !- DayType List 1
 ExteriorLightingProfileDay; !- Schedule:Day Name 1

!- ===== ALL OBJECTS IN CLASS: SCHEDULE:YEAR =====

Schedule:Year,
 S-Lighting, !- Name
 fraction, !- Schedule Type Limits Name
 S-Lighting-week, !- Schedule:Week Name 1
 1, !- Start Month 1
 1, !- Start Day 1
 12, !- End Month 1
 31; !- End Day 1

Schedule:Year,
 Occupancy, !- Name
 Fraction, !- Schedule Type Limits Name
 OccupancyWeek, !- Schedule:Week Name 1
 1, !- Start Month 1
 1, !- Start Day 1
 12, !- End Month 1
 31; !- End Day 1

Schedule:Year,
 LightingProfile, !- Name
 Fraction, !- Schedule Type Limits Name
 LightingProfileWeek, !- Schedule:Week Name 1
 1, !- Start Month 1
 1, !- Start Day 1
 12, !- End Month 1
 31; !- End Day 1

```

Schedule:Year,
  LightingProfile_EELighting, !- Name
  Fraction,                !- Schedule Type Limits Name
  LightingProfileWeek_EELighting, !- Schedule:Week Name 1
  1,                        !- Start Month 1
  1,                        !- Start Day 1
  12,                       !- End Month 1
  31;                       !- End Day 1

```

```

Schedule:Year,
  ExteriorLightingProfile, !- Name
  Fraction,                !- Schedule Type Limits Name
  ExteriorLightingProfileWeek, !- Schedule:Week Name 1
  1,                        !- Start Month 1
  1,                        !- Start Day 1
  12,                       !- End Month 1
  31;                       !- End Day 1

```

```

Schedule:Year,
  Refrigerator,           !- Name
  Fraction,                !- Schedule Type Limits Name
  RefrigeratorWeek,       !- Schedule:Week Name 1
  1,                        !- Start Month 1
  1,                        !- Start Day 1
  12,                       !- End Month 1
  31;                       !- End Day 1

```

```

Schedule:Year,
  MiscPlugLoad,           !- Name
  Fraction,                !- Schedule Type Limits Name
  MiscPlugLoadWeek,       !- Schedule:Week Name 1
  1,                        !- Start Month 1
  1,                        !- Start Day 1
  12,                       !- End Month 1
  31;                       !- End Day 1

```

```

Schedule:Year,
  CookingRange,           !- Name
  Fraction,                !- Schedule Type Limits Name
  CookingRangeWeek,       !- Schedule:Week Name 1
  1,                        !- Start Month 1
  1,                        !- Start Day 1
  12,                       !- End Month 1
  31;                       !- End Day 1

```

```

Schedule:Year,
  dhw_sch,                !- Name
  fraction,                !- Schedule Type Limits Name

```

```

dhw_profile_week,      !- Schedule:Week Name 1
1,                      !- Start Month 1
1,                      !- Start Day 1
12,                     !- End Month 1
31;                     !- End Day 1

Schedule:Year,
  Dishwasher,          !- Name
  Fraction,            !- Schedule Type Limits Name
  DishwasherWeek,      !- Schedule:Week Name 1
  1,                    !- Start Month 1
  1,                    !- Start Day 1
  12,                   !- End Month 1
  31;                   !- End Day 1

Schedule:Year,
  ClothesWasher,       !- Name
  Fraction,            !- Schedule Type Limits Name
  ClothesWasherWeek,   !- Schedule:Week Name 1
  1,                    !- Start Month 1
  1,                    !- Start Day 1
  12,                   !- End Month 1
  31;                   !- End Day 1

Schedule:Year,
  ClothesDryer,        !- Name
  Fraction,            !- Schedule Type Limits Name
  ClothesDryerWeek,    !- Schedule:Week Name 1
  1,                    !- Start Month 1
  1,                    !- Start Day 1
  12,                   !- End Month 1
  31;                   !- End Day 1

Schedule:Year,
  Sinks,               !- Name
  Fraction,            !- Schedule Type Limits Name
  SinksWeek,           !- Schedule:Week Name 1
  1,                    !- Start Month 1
  1,                    !- Start Day 1
  12,                   !- End Month 1
  31;                   !- End Day 1

Schedule:Year,
  Showers,             !- Name
  Fraction,            !- Schedule Type Limits Name
  ShowersWeek,         !- Schedule:Week Name 1
  1,                    !- Start Month 1
  1,                    !- Start Day 1
  12,                   !- End Month 1

```

```

31;                !- End Day 1

Schedule:Year,
  Baths,           !- Name
  Fraction,         !- Schedule Type Limits Name
  BathsWeek,       !- Schedule:Week Name 1
  1,               !- Start Month 1
  1,               !- Start Day 1
  12,              !- End Month 1
  31;              !- End Day 1

Schedule:Year,
  DHWDist,         !- Name
  Fraction,         !- Schedule Type Limits Name
  DHWDistWeek,     !- Schedule:Week Name 1
  1,               !- Start Month 1
  1,               !- Start Day 1
  12,              !- End Month 1
  31;              !- End Day 1

Schedule:Year,
  Occupancy_sch,   !- Name
  Fraction,         !- Schedule Type Limits Name
  OccupancyWeek,   !- Schedule:Week Name 1
  1,               !- Start Month 1
  1,               !- Start Day 1
  12,              !- End Month 1
  31;              !- End Day 1

Schedule:Year,
  Lighting_sch,    !- Name
  Fraction,         !- Schedule Type Limits Name
  LightingProfileWeek, !- Schedule:Week Name 1
  1,               !- Start Month 1
  1,               !- Start Day 1
  12,              !- End Month 1
  31;              !- End Day 1

!- ===== ALL OBJECTS IN CLASS: SCHEDULE:COMPACT =====

Schedule:Compact,
  activity_sch,    !- Name
  any number,      !- Schedule Type Limits Name
  Through: 12/31,  !- Field 1
  For: AllDays,    !- Field 2
  Until: 24:00,    !- Field 3
  117.28;          !- Field 4

```

Schedule:Compact,
 inf_sch, !- Name
 any number, !- Schedule Type Limits Name
 Through: 12/31, !- Field 1
 For: AllDays, !- Field 2
 Until: 24:00, !- Field 3
 1; !- Field 4

Schedule:Compact,
 zone_control_type, !- Name
 control_type, !- Schedule Type Limits Name
 Through: 12/31, !- Field 1
 For: AllDays, !- Field 2
 Until 24:00, !- Field 3
 4; !- Field 4

Schedule:Compact,
 window_shades, !- Name
 fraction, !- Schedule Type Limits Name
 Through: 12/31, !- Field 1
 For: AllDays, !- Field 2
 Until 24:00, !- Field 3
 0.8675; !- Field 4

Schedule:Compact,
 shade_drapes, !- Name
 any number, !- Schedule Type Limits Name
 Through: 5/30, !- Field 1
 For: AllDays, !- Field 2
 Until 24:00, !- Field 3
 0.85, !- Field 4
 Through: 8/31, !- Field 5
 For: AllDays, !- Field 6
 Until 24:00, !- Field 7
 0.7, !- Field 8
 Through: 12/31, !- Field 9
 For: AllDays, !- Field 10
 Until: 24:00, !- Field 11
 0.85; !- Field 12

Schedule:Compact,
 dhw_setpt, !- Name
 Temperature, !- Schedule Type Limits Name
 Through: 12/31, !- Field 1
 For: AllDays, !- Field 2
 Until 24:00, !- Field 3
 48; !- Field 4

Schedule:Compact,

```

Supply-Air-Temp-Sch,    !- Name
Temperature,            !- Schedule Type Limits Name
Through: 12/31,        !- Field 1
For: AllDays,          !- Field 2
Until: 24:00,          !- Field 3
12;                    !- Field 4

Schedule:Compact,
  always_avail,        !- Name
  On/Off,              !- Schedule Type Limits Name
  Through: 12/31,      !- Field 1
  For: AllDays,        !- Field 2
  Until: 24:00,        !- Field 3
  1;                  !- Field 4

Schedule:Compact,
  always_off,          !- Name
  On/Off,              !- Schedule Type Limits Name
  Through: 12/31,      !- Field 1
  For: AllDays,        !- Field 2
  Until: 24:00,        !- Field 3
  0;                  !- Field 4

Schedule:Compact,
  heating_sch,         !- Name
  Temperature,         !- Schedule Type Limits Name
  Through: 12/31,      !- Field 1
  For: AllDays,        !- Field 2
  Until: 24:00,        !- Field 3
  22.22;              !- Field 4

Schedule:Compact,
  cooling_sch,          !- Name
  Temperature,         !- Schedule Type Limits Name
  Through: 12/31,      !- Field 1
  For: AllDays,        !- Field 2
  Until: 24:00,        !- Field 3
  23.88;              !- Field 4

Schedule:Compact,
  fan_cycle,           !- Name
  any number,          !- Schedule Type Limits Name
  Through: 12/31,      !- Field 1
  For: AllDays,        !- Field 2
  Until: 24:00,        !- Field 3
  0;                  !- Field 4

!- ===== ALL OBJECTS IN CLASS: SCHEDULE:CONSTANT =====

```

Schedule:Constant,
DWWaterTempSchedule, !- Name
Temperature, !- Schedule Type Limits Name
48.88888888888889; !- Hourly Value

Schedule:Constant,
CWWaterTempSchedule, !- Name
Temperature, !- Schedule Type Limits Name
48.88888888888889; !- Hourly Value

Schedule:Constant,
SinkSensSchedule, !- Name
Fraction, !- Schedule Type Limits Name
0.687777777777778; !- Hourly Value

Schedule:Constant,
SinkLatSchedule, !- Name
Fraction, !- Schedule Type Limits Name
0.312222222222222; !- Hourly Value

Schedule:Constant,
ShowerSensSchedule, !- Name
Fraction, !- Schedule Type Limits Name
0.51280276816609; !- Hourly Value

Schedule:Constant,
ShowerLatSchedule, !- Name
Fraction, !- Schedule Type Limits Name
0.48719723183391; !- Hourly Value

Schedule:Constant,
BathSensSchedule, !- Name
Fraction, !- Schedule Type Limits Name
1; !- Hourly Value

Schedule:Constant,
BathLatSchedule, !- Name
Fraction, !- Schedule Type Limits Name
0; !- Hourly Value

Schedule:Constant,
SSBWaterTempSchedule, !- Name
Temperature, !- Schedule Type Limits Name
40.5555555555556; !- Hourly Value

Schedule:Constant,
WaterHeaterSP1Schedule, !- Name
Temperature, !- Schedule Type Limits Name

```

48.8888888888889;    !- Hourly Value

Schedule:Constant,
  WaterHeaterSP2Schedule, !- Name
  Temperature,          !- Schedule Type Limits Name
  40.5555555555556;    !- Hourly Value

Schedule:Constant,
  DHWSupplySetpoint,    !- Name
  Temperature,          !- Schedule Type Limits Name
  48.8888888888889;    !- Hourly Value

Schedule:Constant,
  DOASHightemp,         !- Name
  Temperature,          !- Schedule Type Limits Name
  200;                  !- Hourly Value

Schedule:Constant,
  DOASLowtemp,          !- Name
  Temperature,          !- Schedule Type Limits Name
  -60;                  !- Hourly Value

!- ===== ALL OBJECTS IN CLASS: MATERIAL =====

Material,
  wall_consol_layer,    !- Name
  Rough,                !- Roughness
  0.0889,               !- Thickness {m}
  0.05966275,          !- Conductivity {W/m-K}
  120.801,              !- Density {kg/m3}
  1036.25775;          !- Specific Heat {J/kg-K}

Material,
  sheathing_consol_layer, !- Name
  Rough,                !- Roughness
  0.0127,               !- Thickness {m}
  0.0940184,           !- Conductivity {W/m-K}
  685.008,              !- Density {kg/m3}
  1172.332;            !- Specific Heat {J/kg-K}

Material,
  ceil_consol_layer,    !- Name
  Rough,                !- Roughness
  0.266353710534876,   !- Thickness {m}
  0.0617176,           !- Conductivity {W/m-K}
  41.9286,              !- Density {kg/m3}
  776.25126;           !- Specific Heat {J/kg-K}

```


Material,
 floor_consol_layer, !- Name
 Rough, !- Roughness
 0.0889, !- Thickness {m}
 0.0485233, !- Conductivity {W/m-K}
 55.074, !- Density {kg/m3}
 916.9311; !- Specific Heat {J/kg-K}

Material,
 bsmtwall_consol_layer, !- Name
 Rough, !- Roughness
 0.000254, !- Thickness {m}
 0.05966275, !- Conductivity {W/m-K}
 120.801, !- Density {kg/m3}
 1036.25775; !- Specific Heat {J/kg-K}

Material,
 crawlwall_consol_layer, !- Name
 Rough, !- Roughness
 0.000254, !- Thickness {m}
 0.05966275, !- Conductivity {W/m-K}
 120.801, !- Density {kg/m3}
 1036.25775; !- Specific Heat {J/kg-K}

Material,
 Very High Reflectivity Surface, !- Name
 Smooth, !- Roughness
 0.0005, !- Thickness {m}
 237, !- Conductivity {W/m-K}
 2702, !- Density {kg/m3}
 903, !- Specific Heat {J/kg-K}
 0.90, !- Thermal Absorptance
 0.05, !- Solar Absorptance
 0.05; !- Visible Absorptance

Material,
 GypsumBoard-5/16in, !- Name
 Rough, !- Roughness
 7.93953E-03, !- Thickness {m}
 0.1602906, !- Conductivity {W/m-K}
 801, !- Density {kg/m3}
 465.2, !- Specific Heat {J/kg-K}
 0.9, !- Thermal Absorptance
 0.4, !- Solar Absorptance
 0.1; !- Visible Absorptance

Material,
 CopperPipe, !- Name
 MediumRough, !- Roughness

1.90500386169072E-02, !- Thickness {m}
 401, !- Conductivity {W/m-K}
 2243.000, !- Density {kg/m3}
 837.0000, !- Specific Heat {J/kg-K}
 0.9000000, !- Thermal Absorptance
 0.6500000, !- Solar Absorptance
 0.6500000; !- Visible Absorptance

Material,

F08 Metal surface, !- Name
 Smooth, !- Roughness
 0.0008, !- Thickness {m}
 45.28, !- Conductivity {W/m-K}
 7824, !- Density {kg/m3}
 500; !- Specific Heat {J/kg-K}

Material,

Concrete_4in, !- Name
 Rough, !- Roughness
 0.1014984, !- Thickness {m}
 1.312098, !- Conductivity {W/m-K}
 2242.8, !- Density {kg/m3}
 465.2; !- Specific Heat {J/kg-K}

Material,

Asphalt_shingle, !- Name
 MediumRough, !- Roughness
 6.33985285170672E-03, !- Thickness {m}
 0.08186, !- Conductivity {W/m-K}
 1121.2917044623, !- Density {kg/m3}
 1255.20000949809, !- Specific Heat {J/kg-K}
 , !- Thermal Absorptance
 0.75; !- Solar Absorptance

Material,

Wood_shingle, !- Name
 MediumSmooth, !- Roughness
 1.27000257446048E-02, !- Thickness {m}
 0.11388, !- Conductivity {W/m-K}
 426.090847695673, !- Density {kg/m3}
 1631.76001234752; !- Specific Heat {J/kg-K}

Material,

Slate_shingle, !- Name
 MediumSmooth, !- Roughness
 1.27000257446048E-02, !- Thickness {m}
 1.44219, !- Conductivity {W/m-K}
 1601.845292089, !- Density {kg/m3}
 1255.20000949809; !- Specific Heat {J/kg-K}

Material,
 Stucco_1in, !- Name
 MediumSmooth, !- Roughness
 2.54000514892096E-02, !- Thickness {m}
 1.39892, !- Conductivity {W/m-K}
 1922.2143505068, !- Density {kg/m3}
 878.640006648665; !- Specific Heat {J/kg-K}

Material,
 Drywall_1/2in, !- Name
 MediumSmooth, !- Roughness
 1.27000257446048E-02, !- Thickness {m}
 0.16009, !- Conductivity {W/m-K}
 800.922646044499, !- Density {kg/m3}
 1087.84000823168; !- Specific Heat {J/kg-K}

Material,
 OSB_5/8in, !- Name
 MediumSmooth, !- Roughness
 0.015875032180756, !- Thickness {m}
 0.1163, !- Conductivity {W/m-K}
 544.627399310259, !- Density {kg/m3}
 1213.36000918149; !- Specific Heat {J/kg-K}

Material,
 OSB_5/16in, !- Name
 MediumSmooth, !- Roughness
 7.95E-03, !- Thickness {m}
 0.1163, !- Conductivity {W/m-K}
 544.627399310259, !- Density {kg/m3}
 1213.36000918149; !- Specific Heat {J/kg-K}

Material,
 Blown_R30, !- Name
 MediumRough, !- Roughness
 0.212598430964684, !- Thickness {m}
 0.04119, !- Conductivity {W/m-K}
 9.61107175253399, !- Density {kg/m3}
 836.800006332062; !- Specific Heat {J/kg-K}

Material,
 Blown_R30_top, !- Name
 MediumRough, !- Roughness
 0.117348237880148, !- Thickness {m}
 0.04119, !- Conductivity {W/m-K}
 9.61107175253399, !- Density {kg/m3}
 836.800006332062, !- Specific Heat {J/kg-K}
 0.9, !- Thermal Absorptance

0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,

Plywood_3/4in, !- Name
Rough, !- Roughness
0.01905, !- Thickness {m}
0.1154577, !- Conductivity {W/m-K}
544.68, !- Density {kg/m3}
674.54, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,

Batt_R19, !- Name
MediumRough, !- Roughness
2.54000514892096E-02, !- Thickness {m}
3.47522010738099E-03, !- Conductivity {W/m-K}
9.61107175253399, !- Density {kg/m3}
836.800006332062, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,

Lumber_2x4, !- Name
Rough, !- Roughness
0.0890016, !- Thickness {m}
0.1154577, !- Conductivity {W/m-K}
512.64, !- Density {kg/m3}
767.58, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,

Carpet_n_pad, !- Name
MediumSmooth, !- Roughness
2.54000514892096E-02, !- Thickness {m}
6.01314018580031E-02, !- Conductivity {W/m-K}
32.03690584178, !- Density {kg/m3}
836.800006332062, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,

Batt_R13, !- Name

MediumRough, !- Roughness
0.0889, !- Thickness {m}
0.03876, !- Conductivity {W/m-K}
9.61107175253399, !- Density {kg/m3}
836.800006332062, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,
OSB_1/2in, !- Name
MediumSmooth, !- Roughness
1.27000257446048E-02, !- Thickness {m}
0.1163, !- Conductivity {W/m-K}
544.627399310259, !- Density {kg/m3}
1213.36000918149, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,
soil_12in, !- Name
Rough, !- Roughness
0.3048, !- Thickness {m}
1.731, !- Conductivity {W/m-K}
1842.3, !- Density {kg/m3}
232.6, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.7, !- Solar Absorptance
0.7; !- Visible Absorptance

Material,
door_const, !- Name
Smooth, !- Roughness
0.031702180114817, !- Thickness {m}
0.0720096, !- Conductivity {W/m-K}
512.64, !- Density {kg/m3}
767.58; !- Specific Heat {J/kg-K}

Material,
GypsumBoard-1_2in, !- Name
Rough, !- Roughness
0.01271016, !- Thickness {m}
0.1602906, !- Conductivity {W/m-K}
801, !- Density {kg/m3}
465.2, !- Specific Heat {J/kg-K}
0.9, !- Thermal Absorptance
0.4, !- Solar Absorptance
0.1; !- Visible Absorptance

Material,
 Std Wood 6inch, !- Name
 MediumSmooth, !- Roughness
 0.15, !- Thickness {m}
 0.12, !- Conductivity {W/m-K}
 540.0000, !- Density {kg/m3}
 1210, !- Specific Heat {J/kg-K}
 0.9000000, !- Thermal Absorptance
 0.7000000, !- Solar Absorptance
 0.7000000; !- Visible Absorptance

Material,
 Pipe Insulation, !- Name
 VeryRough, !- Roughness
 0.0127032520325203, !- Thickness {m}
 0.03317175, !- Conductivity {W/m-K}
 91.0, !- Density {kg/m3}
 836.0, !- Specific Heat {J/kg-K}
 0.9, !- Thermal Absorptance
 0.5, !- Solar Absorptance
 0.5; !- Visible Absorptance

!- ===== ALL OBJECTS IN CLASS: MATERIAL:NOMASS =====

Material:NoMass,
 Manf_wall_airgap, !- Name
 Smooth, !- Roughness
 0.12; !- Thermal Resistance {m2-K/W}

Material:NoMass,
 Bldg_paper_felt, !- Name
 Smooth, !- Roughness
 1.05666113069662E-02; !- Thermal Resistance {m2-K/W}

Material:NoMass,
 R_high, !- Name
 MediumRough, !- Roughness
 177; !- Thermal Resistance {m2-K/W}

!- ===== ALL OBJECTS IN CLASS: MATERIAL:AIRGAP =====

Material:AirGap,
 Air_4_in_vert, !- Name
 0.158499169604493; !- Thermal Resistance {m2-K/W}

```
!- ===== ALL OBJECTS IN CLASS: WINDOWMATERIAL:SIMPLEGLAZINGSYSTEM
=====
```

```
WindowMaterial:SimpleGlazingSystem,
Glass,           !- Name
2.8393,          !- U-Factor {W/m2-K}
0.25,            !- Solar Heat Gain Coefficient
0.88;            !- Visible Transmittance
```

```
!- ===== ALL OBJECTS IN CLASS: WINDOWMATERIAL:GLAZING =====
```

```
WindowMaterial:Glazing,
Clear Acrylic Plastic, !- Name
SpectralAverage,       !- Optical Data Type
,                      !- Window Glass Spectral Data Set Name
0.003,                !- Thickness {m}
0.92,                 !- Solar Transmittance at Normal Incidence
0.05,                 !- Front Side Solar Reflectance at Normal Incidence
0.05,                 !- Back Side Solar Reflectance at Normal Incidence
0.92,                 !- Visible Transmittance at Normal Incidence
0.05,                 !- Front Side Visible Reflectance at Normal Incidence
0.05,                 !- Back Side Visible Reflectance at Normal Incidence
0.00,                 !- Infrared Transmittance at Normal Incidence
0.90,                 !- Front Side Infrared Hemispherical Emissivity
0.90,                 !- Back Side Infrared Hemispherical Emissivity
0.90;                 !- Conductivity {W/m-K}
```

```
WindowMaterial:Glazing,
Diffusing Acrylic Plastic, !- Name
SpectralAverage,          !- Optical Data Type
,                          !- Window Glass Spectral Data Set Name
0.0022,                  !- Thickness {m}
0.90,                    !- Solar Transmittance at Normal Incidence
0.08,                    !- Front Side Solar Reflectance at Normal Incidence
0.08,                    !- Back Side Solar Reflectance at Normal Incidence
0.90,                    !- Visible Transmittance at Normal Incidence
0.08,                    !- Front Side Visible Reflectance at Normal Incidence
0.08,                    !- Back Side Visible Reflectance at Normal Incidence
0.00,                    !- Infrared Transmittance at Normal Incidence
0.90,                    !- Front Side Infrared Hemispherical Emissivity
0.90,                    !- Back Side Infrared Hemispherical Emissivity
0.90;                    !- Conductivity {W/m-K}
```

```
!- ===== ALL OBJECTS IN CLASS: WINDOWMATERIAL:BLIND =====
```

```
!*** Properties for blinds taken from E+ dataset for 'Blinds with Medium Reflectivity Slats'***
!
```

WindowMaterial:Blind,

```
int_blind,      !- Name
Horizontal,     !- Slat Orientation
0.025,          !- Slat Width {m}
0.01875,        !- Slat Separation {m}
0.001,          !- Slat Thickness {m}
45.0,           !- Slat Angle {deg}
221,            !- Slat Conductivity {W/m-K}
0.0,            !- Slat Beam Solar Transmittance
0.5,            !- Front Side Slat Beam Solar Reflectance
0.5,            !- Back Side Slat Beam Solar Reflectance
0.0,            !- Slat Diffuse Solar Transmittance
0.5,            !- Front Side Slat Diffuse Solar Reflectance
0.5,            !- Back Side Slat Diffuse Solar Reflectance
0.0,            !- Slat Beam Visible Transmittance
0.5,            !- Front Side Slat Beam Visible Reflectance
0.5,            !- Back Side Slat Beam Visible Reflectance
0.0,            !- Slat Diffuse Visible Transmittance
0.5,            !- Front Side Slat Diffuse Visible Reflectance
0.5,            !- Back Side Slat Diffuse Visible Reflectance
0.0,            !- Slat Infrared Hemispherical Transmittance
0.9,            !- Front Side Slat Infrared Hemispherical Emissivity
0.9,            !- Back Side Slat Infrared Hemispherical Emissivity
0.050,          !- Blind to Glass Distance {m}
0.5,            !- Blind Top Opening Multiplier
0.5,            !- Blind Bottom Opening Multiplier
0.5,            !- Blind Left Side Opening Multiplier
0.5,            !- Blind Right Side Opening Multiplier
,               !- Minimum Slat Angle {deg}
;               !- Maximum Slat Angle {deg}
```

!- ===== ALL OBJECTS IN CLASS: CONSTRUCTION =====

Construction,

```
InteriorFurnishings, !- Name
Std Wood 6inch;      !- Outside Layer
```

Construction,

```
ceiling-floor-layer, !- Name
Lumber_2x4;           !- Outside Layer
```

Construction,

```
Exterior Floor,      !- Name
floor_consol_layer,  !- Outside Layer
Plywood_3/4in,       !- Layer 2
Carpet_n_pad;        !- Layer 3
```

Construction,

Interior Floor, !- Name
 Plywood_3/4in, !- Outside Layer
 Carpet_n_pad; !- Layer 2

Construction,
 Exterior Wall, !- Name
 Stucco_1in, !- Outside Layer
 Bldg_paper_felt, !- Layer 2
 sheathing_consol_layer, !- Layer 3
 OSB_5/8in, !- Layer 4
 wall_consol_layer, !- Layer 5
 Drywall_1/2in; !- Layer 6

Construction,
 Interior Ceiling, !- Name
 ceil_consol_layer, !- Outside Layer
 Drywall_1/2in; !- Layer 2

Construction,
 attic floor, !- Name
 Drywall_1/2in, !- Outside Layer
 ceil_consol_layer; !- Layer 2

Construction,
 fndn_roof, !- Name
 Carpet_n_pad, !- Outside Layer
 Plywood_3/4in, !- Layer 2
 ceil_consol_layer; !- Layer 3

Construction,
 interiorwall, !- Name
 Drywall_1/2in, !- Outside Layer
 OSB_5/8in, !- Layer 2
 Drywall_1/2in; !- Layer 3

Construction,
 Interior Wall, !- Name
 Drywall_1/2in, !- Outside Layer
 Air_4_in_vert, !- Layer 2
 Drywall_1/2in; !- Layer 3

Construction,
 Exterior Roof, !- Name
 Asphalt_shingle, !- Outside Layer
 OSB_1/2in; !- Layer 2

Construction,
 Exterior Window, !- Name
 Glass; !- Outside Layer

Construction,
 Interior Window, !- Name
 Glass; !- Outside Layer

Construction,
 Exterior Door, !- Name
 door_const; !- Outside Layer

Construction,
 Interior Door, !- Name
 door_const; !- Outside Layer

Construction,
 Gable_end, !- Name
 Stucco_1in, !- Outside Layer
 Bldg_paper_felt, !- Layer 2
 OSB_5/8in, !- Layer 3
 Air_4_in_vert, !- Layer 4
 Drywall_1/2in; !- Layer 5

Construction,
 crawl_floor, !- Name
 R_high, !- Outside Layer
 soil_12in; !- Layer 2

Construction,
 window_w_blinds, !- Name
 Glass, !- Outside Layer
 int_blind; !- Layer 2

Construction,
 Insulated Pipe, !- Name
 Pipe Insulation, !- Outside Layer
 CopperPipe; !- Layer 2

Construction,
 Plain Pipe, !- Name
 CopperPipe; !- Outside Layer

Construction,
 TDD Pipe, !- Name
 Very High Reflectivity Surface; !- Outside Layer

Construction,
 TDD Dome, !- Name
 Clear Acrylic Plastic; !- Outside Layer

Construction,

```

TDD Diffuser,      !- Name
Diffusing Acrylic Plastic; !- Outside Layer

Construction,
  crawl wall,      !- Name
  Concrete_4in;    !- Outside Layer

!- ===== ALL OBJECTS IN CLASS: GLOBALGEOMETRYRULES =====

GlobalGeometryRules,
  LowerLeftCorner,  !- Starting Vertex Position
  Counterclockwise, !- Vertex Entry Direction
  Relative;        !- Coordinate System

!- ===== ALL OBJECTS IN CLASS: ZONE =====

Zone,
  living_unit1,    !- Name
  0.0,             !- Direction of Relative North {deg}
  0.0,             !- X Origin {m}
  0.0,             !- Y Origin {m}
  0.0,             !- Z Origin {m}
  ,               !- Type
  1;              !- Multiplier

Zone,
  attic_unit1,     !- Name
  0.0,             !- Direction of Relative North {deg}
  0.0,             !- X Origin {m}
  0.0,             !- Y Origin {m}
  0.0,             !- Z Origin {m}
  ,               !- Type
  1;              !- Multiplier

Zone,
  crawlspace_unit1, !- Name
  0.0,             !- Direction of Relative North {deg}
  0.0,             !- X Origin {m}
  0.0,             !- Y Origin {m}
  0.0,             !- Z Origin {m}
  ,               !- Type
  1;              !- Multiplier

!- ===== ALL OBJECTS IN CLASS: BUILDINGSURFACE:DETAILED =====

BuildingSurface:Detailed,

```

Inter zone floor 1, !- Name
 Floor, !- Surface Type
 Interior Floor, !- Construction Name
 living_unit1, !- Zone Name
 Adiabatic, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 0.0, !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 2.74390243902439, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 2.74390243902439, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 2.74390243902439, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 2.74390243902439; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,
 ceiling_unit1, !- Name
 Ceiling, !- Surface Type
 Interior Ceiling, !- Construction Name
 living_unit1, !- Zone Name
 Zone, !- Outside Boundary Condition
 attic_unit1, !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 0.0, !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 5.33536585365854, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 5.33536585365854, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 5.33536585365854, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 5.33536585365854; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,
 Roof_front_unit1, !- Name

Roof, !- Surface Type
 Exterior Roof, !- Construction Name
 attic_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 0.0, !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 5.33536585365854, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 5.33536585365854, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 4.57855748470485, !- Vertex 3 Y-coordinate {m}
 6.70911265750324, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 4.57855748470485, !- Vertex 4 Y-coordinate {m}
 6.70911265750324; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Roof_back_unit1, !- Name
 Roof, !- Surface Type
 Exterior Roof, !- Construction Name
 attic_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 0.0, !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 5.33536585365854, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 5.33536585365854, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 4.57855748470485, !- Vertex 3 Y-coordinate {m}
 6.70911265750324, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 4.57855748470485, !- Vertex 4 Y-coordinate {m}
 6.70911265750324; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Roof_right_unit1, !- Name
 Wall, !- Surface Type

Gable_end, !- Construction Name
 attic_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 3, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 5.33536585365854, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 5.33536585365854, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 4.57855748470485, !- Vertex 3 Y-coordinate {m}
 6.70911265750324; !- Vertex 3 Z-coordinate {m}

BuildingSurface:Detailed,
 Roof_left_unit1, !- Name
 Wall, !- Surface Type
 Gable_end, !- Construction Name
 attic_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 3, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 5.33536585365854, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 5.33536585365854, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 4.57855748470485, !- Vertex 3 Y-coordinate {m}
 6.70911265750324; !- Vertex 3 Z-coordinate {m}

BuildingSurface:Detailed,
 Extfloor_unit1, !- Name
 Floor, !- Surface Type
 Interior Floor, !- Construction Name
 crawlspace_unit1, !- Zone Name
 GroundSlabPreprocessorAverage, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 0.0, !- View Factor to Ground

4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 -0.457317073170732, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 -0.457317073170732, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 -0.457317073170732, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 -0.457317073170732; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_Idf_1.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 0.152439024390244, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 0.152439024390244, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 2.74390243902439, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 2.74390243902439; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_sdr_1.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices

12.1789629093149, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 0.152439024390244, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 0.152439024390244, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 2.74390243902439, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 2.74390243902439; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_ldb_1.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 0.152439024390244, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 0.152439024390244, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 2.74390243902439, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 2.74390243902439; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_sdl_1.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}

9.15711496940971, !- Vertex 1 Y-coordinate {m}
 0.152439024390244, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 0.152439024390244, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 2.74390243902439, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 2.74390243902439; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_ldf_2.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 2.74390243902439, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 2.74390243902439, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 5.33536585365854, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 5.33536585365854; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_sdr_2.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}

2.74390243902439, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 2.74390243902439, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 5.33536585365854, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 5.33536585365854; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_ldb_2.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 2.74390243902439, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 2.74390243902439, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 5.33536585365854, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 5.33536585365854; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Wall_sdl_2.unit1, !- Name
 Wall, !- Surface Type
 Exterior Wall, !- Construction Name
 living_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 2.74390243902439, !- Vertex 1 Z-coordinate {m}

0, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 2.74390243902439, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 5.33536585365854, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 5.33536585365854; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

Floor_unit1, !- Name
 Floor, !- Surface Type
 Exterior Floor, !- Construction Name
 living_unit1, !- Zone Name
 Zone, !- Outside Boundary Condition
 crawlspace_unit1, !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 0.0, !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 0.152439024390244, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 0.152439024390244, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 0.152439024390244, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 0.152439024390244; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

BGWall_upper_ldf, !- Name
 Wall, !- Surface Type
 Crawl Wall, !- Construction Name
 crawlspace_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 0, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}

0, !- Vertex 2 Y-coordinate {m}
 0, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 0.152439024390244, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 0.152439024390244; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,
 BGWall_lower_ldf, !- Name
 Wall, !- Surface Type
 Crawl Wall, !- Construction Name
 crawlspace_unit1, !- Zone Name
 GroundSlabPreprocessorAverage, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 -0.457317073170732, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 -0.457317073170732, !- Vertex 2 Z-coordinate {m}
 12.1789629093149, !- Vertex 3 X-coordinate {m}
 0, !- Vertex 3 Y-coordinate {m}
 0, !- Vertex 3 Z-coordinate {m}
 0, !- Vertex 4 X-coordinate {m}
 0, !- Vertex 4 Y-coordinate {m}
 0; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,
 BGWall_upper_sdr, !- Name
 Wall, !- Surface Type
 Crawl Wall, !- Construction Name
 crawlspace_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 0, !- Vertex 1 Y-coordinate {m}
 0, !- Vertex 1 Z-coordinate {m}
 12.1789629093149, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}

```

0,                !- Vertex 2 Z-coordinate {m}
12.1789629093149,    !- Vertex 3 X-coordinate {m}
9.15711496940971,    !- Vertex 3 Y-coordinate {m}
0.152439024390244,    !- Vertex 3 Z-coordinate {m}
12.1789629093149,    !- Vertex 4 X-coordinate {m}
0,                !- Vertex 4 Y-coordinate {m}
0.152439024390244;    !- Vertex 4 Z-coordinate {m}

```

BuildingSurface:Detailed,

```

BGWall_lower_sdr,    !- Name
Wall,                !- Surface Type
Crawl Wall,          !- Construction Name
crawlspace_unit1,    !- Zone Name
GroundSlabPreprocessorAverage, !- Outside Boundary Condition
,                    !- Outside Boundary Condition Object
NoSun,               !- Sun Exposure
NoWind,              !- Wind Exposure
,                    !- View Factor to Ground
4,                   !- Number of Vertices
12.1789629093149,    !- Vertex 1 X-coordinate {m}
0,                   !- Vertex 1 Y-coordinate {m}
-0.457317073170732,  !- Vertex 1 Z-coordinate {m}
12.1789629093149,    !- Vertex 2 X-coordinate {m}
9.15711496940971,    !- Vertex 2 Y-coordinate {m}
-0.457317073170732,  !- Vertex 2 Z-coordinate {m}
12.1789629093149,    !- Vertex 3 X-coordinate {m}
9.15711496940971,    !- Vertex 3 Y-coordinate {m}
0,                   !- Vertex 3 Z-coordinate {m}
12.1789629093149,    !- Vertex 4 X-coordinate {m}
0,                   !- Vertex 4 Y-coordinate {m}
0;                   !- Vertex 4 Z-coordinate {m}

```

BuildingSurface:Detailed,

```

BGWall_upper_ldb,    !- Name
Wall,                !- Surface Type
Crawl Wall,          !- Construction Name
crawlspace_unit1,    !- Zone Name
Outdoors,            !- Outside Boundary Condition
,                    !- Outside Boundary Condition Object
SunExposed,          !- Sun Exposure
WindExposed,         !- Wind Exposure
,                    !- View Factor to Ground
4,                   !- Number of Vertices
12.1789629093149,    !- Vertex 1 X-coordinate {m}
9.15711496940971,    !- Vertex 1 Y-coordinate {m}
0,                   !- Vertex 1 Z-coordinate {m}
0,                   !- Vertex 2 X-coordinate {m}
9.15711496940971,    !- Vertex 2 Y-coordinate {m}
0,                   !- Vertex 2 Z-coordinate {m}

```

0, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 0.152439024390244, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 0.152439024390244; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

BGWall_lower_ldb, !- Name
 Wall, !- Surface Type
 Crawl Wall, !- Construction Name
 crawlspace_unit1, !- Zone Name
 GroundSlabPreprocessorAverage, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 NoSun, !- Sun Exposure
 NoWind, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 12.1789629093149, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 -0.457317073170732, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 9.15711496940971, !- Vertex 2 Y-coordinate {m}
 -0.457317073170732, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}
 9.15711496940971, !- Vertex 3 Y-coordinate {m}
 0, !- Vertex 3 Z-coordinate {m}
 12.1789629093149, !- Vertex 4 X-coordinate {m}
 9.15711496940971, !- Vertex 4 Y-coordinate {m}
 0; !- Vertex 4 Z-coordinate {m}

BuildingSurface:Detailed,

BGWall_upper_sdl, !- Name
 Wall, !- Surface Type
 Crawl Wall, !- Construction Name
 crawlspace_unit1, !- Zone Name
 Outdoors, !- Outside Boundary Condition
 , !- Outside Boundary Condition Object
 SunExposed, !- Sun Exposure
 WindExposed, !- Wind Exposure
 , !- View Factor to Ground
 4, !- Number of Vertices
 0, !- Vertex 1 X-coordinate {m}
 9.15711496940971, !- Vertex 1 Y-coordinate {m}
 0, !- Vertex 1 Z-coordinate {m}
 0, !- Vertex 2 X-coordinate {m}
 0, !- Vertex 2 Y-coordinate {m}
 0, !- Vertex 2 Z-coordinate {m}
 0, !- Vertex 3 X-coordinate {m}

```

0,                !- Vertex 3 Y-coordinate {m}
0.152439024390244,    !- Vertex 3 Z-coordinate {m}
0,                !- Vertex 4 X-coordinate {m}
9.15711496940971,    !- Vertex 4 Y-coordinate {m}
0.152439024390244;    !- Vertex 4 Z-coordinate {m}

```

BuildingSurface:Detailed,

```

BGWall_lower_sdl,    !- Name
Wall,                !- Surface Type
Crawl Wall,          !- Construction Name
crawlspcace_unit1,    !- Zone Name
GroundSlabPreprocessorAverage, !- Outside Boundary Condition
,                    !- Outside Boundary Condition Object
NoSun,               !- Sun Exposure
NoWind,              !- Wind Exposure
,                    !- View Factor to Ground
4,                   !- Number of Vertices
0,                   !- Vertex 1 X-coordinate {m}
9.15711496940971,    !- Vertex 1 Y-coordinate {m}
-0.457317073170732,    !- Vertex 1 Z-coordinate {m}
0,                   !- Vertex 2 X-coordinate {m}
0,                   !- Vertex 2 Y-coordinate {m}
-0.457317073170732,    !- Vertex 2 Z-coordinate {m}
0,                   !- Vertex 3 X-coordinate {m}
0,                   !- Vertex 3 Y-coordinate {m}
0,                   !- Vertex 3 Z-coordinate {m}
0,                   !- Vertex 4 X-coordinate {m}
9.15711496940971,    !- Vertex 4 Y-coordinate {m}
0;                   !- Vertex 4 Z-coordinate {m}

```

!- ===== ALL OBJECTS IN CLASS: WINDOW =====

Window,

```

Window_Idf_1.unit1,    !- Name
Exterior Window,       !- Construction Name
Wall_Idf_1.unit1,      !- Building Surface Name
Shades,                !- Shading Control Name
,                      !- Frame and Divider Name
1,                     !- Multiplier
1.91449814126394,      !- Starting X Coordinate {m}
0.914634146341463,     !- Starting Z Coordinate {m}
2.72034992953739,     !- Length {m}
1.52439024390244;      !- Height {m}

```

Window,

```

Window_Idb_1.unit1,    !- Name
Exterior Window,       !- Construction Name
Wall_Idb_1.unit1,      !- Building Surface Name

```

Shades, !- Shading Control Name
 , !- Frame and Divider Name
 1, !- Multiplier
 1.91449814126394, !- Starting X Coordinate {m}
 0.914634146341463, !- Starting Z Coordinate {m}
 2.72034992953739, !- Length {m}
 1.52439024390244; !- Height {m}

Window,
 Window_sdr_1.unit1, !- Name
 Exterior Window, !- Construction Name
 Wall_sdr_1.unit1, !- Building Surface Name
 Shades, !- Shading Control Name
 , !- Frame and Divider Name
 1, !- Multiplier
 1, !- Starting X Coordinate {m}
 0.914634146341463, !- Starting Z Coordinate {m}
 2.72034992953739, !- Length {m}
 1.52439024390244; !- Height {m}

Window,
 Window_sdl_1.unit1, !- Name
 Exterior Window, !- Construction Name
 Wall_sdl_1.unit1, !- Building Surface Name
 Shades, !- Shading Control Name
 , !- Frame and Divider Name
 1, !- Multiplier
 1, !- Starting X Coordinate {m}
 0.914634146341463, !- Starting Z Coordinate {m}
 2.72034992953739, !- Length {m}
 1.52439024390244; !- Height {m}

Window,
 Window_ldf_2.unit1, !- Name
 Exterior Window, !- Construction Name
 Wall_ldf_2.unit1, !- Building Surface Name
 Shades, !- Shading Control Name
 , !- Frame and Divider Name
 1, !- Multiplier
 1.91449814126394, !- Starting X Coordinate {m}
 0.914634146341463, !- Starting Z Coordinate {m}
 2.72034992953739, !- Length {m}
 1.52439024390244; !- Height {m}

Window,
 Window_ldb_2.unit1, !- Name
 Exterior Window, !- Construction Name
 Wall_ldb_2.unit1, !- Building Surface Name
 Shades, !- Shading Control Name


```
,           !- Frame and Divider Name
1,           !- Multiplier
1.91449814126394,    !- Starting X Coordinate {m}
0.914634146341463,    !- Starting Z Coordinate {m}
2.72034992953739,    !- Length {m}
1.52439024390244;    !- Height {m}
```

```
Window,
  Window_sdr_2.unit1,    !- Name
  Exterior Window,       !- Construction Name
  Wall_sdr_2.unit1,      !- Building Surface Name
  Shades,                !- Shading Control Name
  ,                       !- Frame and Divider Name
  1,                       !- Multiplier
  1,                       !- Starting X Coordinate {m}
  0.914634146341463,     !- Starting Z Coordinate {m}
  2.72034992953739,     !- Length {m}
  1.52439024390244;     !- Height {m}
```

```
Window,
  Window_sdl_2.unit1,    !- Name
  Exterior Window,       !- Construction Name
  Wall_sdl_2.unit1,      !- Building Surface Name
  Shades,                !- Shading Control Name
  ,                       !- Frame and Divider Name
  1,                       !- Multiplier
  1,                       !- Starting X Coordinate {m}
  0.914634146341463,     !- Starting Z Coordinate {m}
  2.72034992953739,     !- Length {m}
  1.52439024390244;     !- Height {m}
```

!- ===== ALL OBJECTS IN CLASS: DOOR =====

```
Door,
  Door_ldf_unit1,        !- Name
  Exterior Door,         !- Construction Name
  Wall_ldf_1.unit1,      !- Building Surface Name
  1,                       !- Multiplier
  0.5,                   !- Starting X Coordinate {m}
  0,                     !- Starting Z Coordinate {m}
  0.914498141263941,     !- Length {m}
  2.13414634146341;     !- Height {m}
```

```
Door,
  Door_ldb_unit1,        !- Name
  Exterior Door,         !- Construction Name
  Wall_ldb_1.unit1,      !- Building Surface Name
  1,                       !- Multiplier
```

0.5, !- Starting X Coordinate {m}
0, !- Starting Z Coordinate {m}
0.914498141263941, !- Length {m}
2.13414634146341; !- Height {m}

!- ===== ALL OBJECTS IN CLASS: WINDOWPROPERTY:SHADINGCONTROL
=====

WindowProperty:ShadingControl,
 Shades, !- Name
 InteriorBlind, !- Shading Type
 window_w_blinds, !- Construction with Shading Name
 OnIfScheduleAllows, !- Shading Control Type
 window_shades, !- Schedule Name
 , !- Setpoint {W/m2, W or deg C}
 Yes, !- Shading Control Is Scheduled
 No, !- Glare Control Is Active
 , !- Shading Device Material Name
 , !- Type of Slat Angle Control for Blinds
 ; !- Slat Angle Schedule Name

!- ===== ALL OBJECTS IN CLASS: INTERNALMASS =====

InternalMass,
 Internalmass_unit1, !- Name
 InteriorFurnishings, !- Construction Name
 living_unit1, !- Zone Name
 9.99586947542338; !- Surface Area {m2}

!- ===== ALL OBJECTS IN CLASS: SHADING:OVERHANG =====

Shading:Overhang,
 Overhang_sdr_1.unit1, !- Name
 Window_sdr_1.unit1, !- Window or Door Name
 0, !- Height above Window or Door {m}
 90, !- Tilt Angle from Window/Door {deg}
 0, !- Left extension from Window/Door Width {m}
 0, !- Right extension from Window/Door Width {m}
 0.0152439024390244; !- Depth {m}

Shading:Overhang,
 Overhang_sdr_2.unit1, !- Name
 Window_sdr_2.unit1, !- Window or Door Name
 0, !- Height above Window or Door {m}
 90, !- Tilt Angle from Window/Door {deg}
 0, !- Left extension from Window/Door Width {m}

```

0,          !- Right extension from Window/Door Width {m}
0.0152439024390244;    !- Depth {m}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:CONTROL =====

GroundHeatTransfer:Control,
  gtp_control,      !- Name
  no,               !- Run Basement Preprocessor
  yes;              !- Run Slab Preprocessor

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:MATERIALS
=====

GroundHeatTransfer:Slab:Materials,
  2,                !- NMAT: Number of materials
  0.16,             !- ALBEDO: Surface Albedo: No Snow
  0.4,              !- ALBEDO: Surface Albedo: Snow
  0.9,              !- EPSLW: Surface Emissivity: No Snow
  0.9,              !- EPSLW: Surface Emissivity: Snow
  0.75,             !- Z0: Surface Roughness: No Snow {cm}
  0.05,             !- Z0: Surface Roughness: Snow {cm}
  6,                !- HIN: Indoor HConv: Downward Flow {W/m2-K}
  9;                !- HIN: Indoor HConv: Upward {W/m2-K}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:MATLPROPS
=====

GroundHeatTransfer:Slab:MatlProps,
  2300,             !- RHO: Slab Material density {kg/m3}
  1200,             !- RHO: Soil Density {kg/m3}
  650,              !- CP: Slab CP {J/kg-K}
  1200,             !- CP: Soil CP {J/kg-K}
  0.9,              !- TCON: Slab k {W/m-K}
  1;                !- TCON: Soil k {W/m-K}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:BOUNDCONDS
=====

GroundHeatTransfer:Slab:BoundConds,
  FALSE,            !- EVTR: Is surface evapotranspiration modeled
  TRUE,             !- FIXBC: is the lower boundary at a fixed temperature
  10,               !- TDEEPin {C}
  FALSE;            !- USRHflag: Is the ground surface h specified by the user?

```

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:BLDGPROPS
=====

GroundHeatTransfer:Slab:BldgProps,
 10, !- IYRS: Number of years to iterate
 0, !- Shape: Slab shape
 4, !- HBLDG: Building height {m}
 22.22, !- TIN1: January Indoor Average Temperature Setpoint {C}
 22.22, !- TIN2: February Indoor Average Temperature Setpoint {C}
 22.22, !- TIN3: March Indoor Average Temperature Setpoint {C}
 22.22, !- TIN4: April Indoor Average Temperature Setpoint {C}
 22.22, !- TIN5: May Indoor Average Temperature Setpoint {C}
 23.88, !- TIN6: June Indoor Average Temperature Setpoint {C}
 23.88, !- TIN7: July Indoor Average Temperature Setpoint {C}
 23.88, !- TIN8: August Indoor Average Temperature Setpoint {C}
 23.88, !- TIN9: September Indoor Average Temperature Setpoint {C}
 22.22, !- TIN10: October Indoor Average Temperature Setpoint {C}
 22.22, !- TIN11: November Indoor Average Temperature Setpoint {C}
 22.22, !- TIN12: December Indoor Average Temperature Setpoint {C}
 0, !- TINamp: Daily Indoor sine wave variation amplitude {deltaC}
 0.1; !- ConvTol: Convergence Tolerance

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:INSULATION
=====

GroundHeatTransfer:Slab:Insulation,
 0, !- RINS: R value of under slab insulation {m2-K/W}
 0, !- DINS: Width of strip of under slab insulation {m}
 0, !- RVINS: R value of vertical insulation {m2-K/W}
 1.5, !- ZVINS: Depth of vertical insulation {m}
 0; !- IVINS: Flag: Is there vertical insulation

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:SLAB:EQUIVALENTSLAB
=====

GroundHeatTransfer:Slab:EquivalentSlab,
 2.8436453177271, !- APRatio: The area to perimeter ratio for this slab {m}
 0.1, !- SLABDEPTH: Thickness of slab on grade {m}
 15, !- CLEARANCE: Distance from edge of slab to domain edge {m}
 15; !- ZCLEARANCE: Distance from bottom of slab to domain bottom {m}

!- ===== ALL OBJECTS IN CLASS:
GROUNDHEATTRANSFER:BASEMENT:SIMPARAMETERS =====

GroundHeatTransfer:Basement:SimParameters,
 0.1, !- F: Multiplier for the ADI solution

15; !- IYRS: Maximum number of yearly iterations:

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:MATLPROPS
=====

GroundHeatTransfer:Basement:MatlProps,
6, !- NMAT: Number of materials in this domain
2243, !- Density for Foundation Wall {kg/m3}
2243, !- density for Floor Slab {kg/m3}
311, !- density for Ceiling {kg/m3}
1500, !- density for Soil {kg/m3}
2000, !- density for Gravel {kg/m3}
449, !- density for Wood {kg/m3}
880, !- Specific heat for foundation wall {J/kg-K}
880, !- Specific heat for floor slab {J/kg-K}
1530, !- Specific heat for ceiling {J/kg-K}
840, !- Specific heat for soil {J/kg-K}
720, !- Specific heat for gravel {J/kg-K}
1530, !- Specific heat for wood {J/kg-K}
1.4, !- Thermal conductivity for foundation wall {W/m-K}
1.4, !- Thermal conductivity for floor slab {W/m-K}
0.09, !- Thermal conductivity for ceiling {W/m-K}
1.1, !- thermal conductivity for soil {W/m-K}
1.9, !- thermal conductivity for gravel {W/m-K}
0.12; !- thermal conductivity for wood {W/m-K}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:INSULATION
=====

GroundHeatTransfer:Basement:Insulation,
0, !- REXT: R Value of any exterior insulation {m2-K/W}
True; !- INSFULL: Flag: Is the wall fully insulated?

!- ===== ALL OBJECTS IN CLASS:
GROUNDHEATTRANSFER:BASEMENT:SURFACEPROPS =====

GroundHeatTransfer:Basement:SurfaceProps,
0.16, !- ALBEDO: Surface albedo for No snow conditions
0.4, !- ALBEDO: Surface albedo for snow conditions
0.94, !- EPSLN: Surface emissivity No Snow
0.86, !- EPSLN: Surface emissivity with Snow
6, !- VEGHT: Surface roughness No snow conditions {cm}
0.25, !- VEGHT: Surface roughness Snow conditions {cm}
False; !- PET: Flag, Potential evapotranspiration on?

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:BLDGDATA
=====

GroundHeatTransfer:Basement:BldgData,
 0.200000006162114, !- DWALL: Wall thickness {m}
 0.243828108701145, !- DSLAB: Floor slab thickness {m}
 0.3, !- DGRAVXY: Width of gravel pit beside basement wall {m}
 0.2, !- DGRAVZN: Gravel depth extending above the floor slab {m}
 0.1; !- DGRAVZP: Gravel depth below the floor slab {m}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:INTERIOR
=====

GroundHeatTransfer:Basement:Interior,
 True, !- COND: Flag: Is the basement conditioned?
 0.92, !- HIN: Downward convection only heat transfer coefficient {W/m2-K}
 4.04, !- HIN: Upward convection only heat transfer coefficient {W/m2-K}
 3.08, !- HIN: Horizontal convection only heat transfer coefficient {W/m2-K}
 6.13, !- HIN: Downward combined (convection and radiation) heat transfer coefficient
 {W/m2-K}
 9.26, !- HIN: Upward combined (convection and radiation) heat transfer coefficient {W/m2-
 K}
 8.29; !- HIN: Horizontal combined (convection and radiation) heat transfer coefficient
 {W/m2-K}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:COMBLDG
=====

GroundHeatTransfer:Basement:ComBldg,
 21, !- January average temperature {C}
 21, !- February average temperature {C}
 21, !- March average temperature {C}
 21, !- April average temperature {C}
 24, !- May average temperature {C}
 24, !- June average temperature {C}
 24, !- July average temperature {C}
 24, !- August average temperature {C}
 24, !- September average temperature {C}
 24, !- October average temperature {C}
 21, !- November average temperature {C}
 21, !- December average temperature {C}
 21; !- Daily variation sine wave amplitude {deltaC}

!- ===== ALL OBJECTS IN CLASS: GROUNDHEATTRANSFER:BASEMENT:EQUIVSLAB
=====

GroundHeatTransfer:Basement:EquivSlab,
 17.1459093179727, !- APRatio: The area to perimeter ratio for this slab {m}
 True; !- EquivSizing: Flag

!- ===== ALL OBJECTS IN CLASS:
 GROUNDHEATTRANSFER:BASEMENT:EQUIVAUTOGRID =====

GroundHeatTransfer:Basement:EquivAutoGrid,
 15, !- CLEARANCE: Distance from outside of wall to edge of 3-D ground domain {m}
 0.1, !- SlabDepth: Thickness of the floor slab {m}
 1.21914054350572; !- BaseDepth: Depth of the basement wall below grade {m}

!- ===== ALL OBJECTS IN CLASS: PEOPLE =====

People,
 people_unit1, !- Name
 living_unit1, !- Zone or ZoneList Name
 occupancy_sch, !- Number of People Schedule Name
 People, !- Number of People Calculation Method
 3, !- Number of People
 , !- People per Zone Floor Area {person/m2}
 , !- Zone Floor Area per Person {m2/person}
 0, !- Fraction Radiant
 autocalculate, !- Sensible Heat Fraction
 activity_sch, !- Activity Level Schedule Name
 , !- Carbon Dioxide Generation Rate {m3/s-W}
 No, !- Enable ASHRAE 55 Comfort Warnings
 ZoneAveraged; !- Mean Radiant Temperature Calculation Type

!- ===== ALL OBJECTS IN CLASS: LIGHTS =====

Lights,
 Living Hardwired Lighting1, !- Name
 living_unit1, !- Zone or ZoneList Name
 LightingProfile_EELighting, !- Schedule Name
 Watts/Area, !- Design Level Calculation Method
 , !- Lighting Level {W}
 1.222, !- Watts per Zone Floor Area {W/m2}
 , !- Watts per Person {W/person}
 0, !- Return Air Fraction
 0.6, !- Fraction Radiant
 0.2, !- Fraction Visible
 0; !- Fraction Replaceable

Lights,
 Living Plug-in Lighting1, !- Name

```

living_unit1,      !- Zone or ZoneList Name
LightingProfile_EELighting, !- Schedule Name
Watts/Area,        !- Design Level Calculation Method
,                  !- Lighting Level {W}
0.478,             !- Watts per Zone Floor Area {W/m2}
,                  !- Watts per Person {W/person}
0,                 !- Return Air Fraction
0.6,               !- Fraction Radiant
0.2,               !- Fraction Visible
0;                 !- Fraction Replaceable

```

!- ===== ALL OBJECTS IN CLASS: ELECTRIC EQUIPMENT =====

```

ElectricEquipment,
  Refrigerator1,    !- Name
  living_unit1,     !- Zone or ZoneList Name
  Refrigerator,     !- Schedule Name
  EquipmentLevel,   !- Design Level Calculation Method
  91.0939726027399, !- Design Level {W}
  ,                 !- Watts per Zone Floor Area {W/m2}
  ,                 !- Watts per Person {W/person}
  0,                !- Fraction Latent
  0,                !- Fraction Radiant
  0,                !- Fraction Lost
  Appl;             !- End-Use Subcategory

```

```

ElectricEquipment,
  Misc Elec Load1, !- Name
  living_unit1,     !- Zone or ZoneList Name
  MiscPlugLoad,     !- Schedule Name
  Watts/Area,       !- Design Level Calculation Method
  ,                 !- Design Level {W}
  2.46,             !- Watts per Zone Floor Area {W/m2}
  ,                 !- Watts per Person {W/person}
  0.0624390461422629, !- Fraction Latent
  0.41190936353998, !- Fraction Radiant
  0.251045347957769, !- Fraction Lost
  Misc;             !- End-Use Subcategory

```

```

ElectricEquipment,
  Clothes Washer1,  !- Name
  living_unit1,     !- Zone or ZoneList Name
  ClothesWasher,    !- Schedule Name
  EquipmentLevel,   !- Design Level Calculation Method
  29.5818312328767, !- Design Level {W}
  ,                 !- Watts per Zone Floor Area {W/m2}
  ,                 !- Watts per Person {W/person}
  0,                !- Fraction Latent

```


0.48, !- Fraction Radiant
 0.2, !- Fraction Lost
 Appl; !- End-Use Subcategory

ElectricEquipment,
 Clothes Dryer1, !- Name
 living_unit1, !- Zone or ZoneList Name
 ClothesDryer, !- Schedule Name
 EquipmentLevel, !- Design Level Calculation Method
 222.112928219178, !- Design Level {W}
 , !- Watts per Zone Floor Area {W/m2}
 , !- Watts per Person {W/person}
 0.05, !- Fraction Latent
 0.09, !- Fraction Radiant
 0.8, !- Fraction Lost
 Appl; !- End-Use Subcategory

ElectricEquipment,
 Dishwasher1, !- Name
 living_unit1, !- Zone or ZoneList Name
 Dishwasher, !- Schedule Name
 EquipmentLevel, !- Design Level Calculation Method
 68.3275708931507, !- Design Level {W}
 , !- Watts per Zone Floor Area {W/m2}
 , !- Watts per Person {W/person}
 0.15, !- Fraction Latent
 0.36, !- Fraction Radiant
 0.25, !- Fraction Lost
 Appl; !- End-Use Subcategory

ElectricEquipment,
 Misc Elec Load21, !- Name
 living_unit1, !- Zone or ZoneList Name
 always_avail, !- Schedule Name
 EquipmentLevel, !- Design Level Calculation Method
 182.521535772157, !- Design Level {W}
 , !- Watts per Zone Floor Area {W/m2}
 , !- Watts per Person {W/person}
 0.0624390461422629, !- Fraction Latent
 0.41190936353998, !- Fraction Radiant
 0.251045347957769, !- Fraction Lost
 Misc; !- End-Use Subcategory

!- ===== ALL OBJECTS IN CLASS: GASEQUIPMENT =====

GasEquipment,
 Cooking Range1, !- Name
 living_unit1, !- Zone or ZoneList Name

CookingRange, !- Schedule Name
 EquipmentLevel, !- Design Level Calculation Method
 248.497534246575, !- Design Level {W}
 , !- Power per Zone Floor Area {W/m2}
 , !- Power per Person {W/Person}
 0.3, !- Fraction Latent
 0.24, !- Fraction Radiant
 0.3, !- Fraction Lost
 , !- Carbon Dioxide Generation Rate {m3/s-W}
 Appl; !- End-Use Subcategory

GasEquipment,
 Misc Gas Load1, !- Name
 living_unit1, !- Zone or ZoneList Name
 MiscPlugLoad, !- Schedule Name
 Watts/Area, !- Design Level Calculation Method
 , !- Design Level {W}
 0.297, !- Power per Zone Floor Area {W/m2}
 , !- Power per Person {W/Person}
 0.0624390461422629, !- Fraction Latent
 0.41190936353998, !- Fraction Radiant
 0.251045347957769, !- Fraction Lost
 , !- Carbon Dioxide Generation Rate {m3/s-W}
 Misc; !- End-Use Subcategory

!- ===== ALL OBJECTS IN CLASS: ZONEINFILTRATION:EFFECTIVELEAKAGEAREA
 =====

ZoneInfiltration:EffectiveLeakageArea,
 Living_ShermanGrimsrud_unit1, !- Name
 living_unit1, !- Zone Name
 always_avail, !- Schedule Name
 601.688097849621, !- Effective Air Leakage Area {cm2}
 0.00029, !- Stack Coefficient
 0.000231; !- Wind Coefficient

ZoneInfiltration:EffectiveLeakageArea,
 AtticVent_unit1, !- Name
 attic_unit1, !- Zone Name
 always_avail, !- Schedule Name
 370, !- Effective Air Leakage Area {cm2}
 0.00029, !- Stack Coefficient
 0.000231; !- Wind Coefficient

ZoneInfiltration:EffectiveLeakageArea,
 CrawlVent_unit1, !- Name
 crawlspace_unit1, !- Zone Name
 always_avail, !- Schedule Name

370, !- Effective Air Leakage Area {cm2}
0.00029, !- Stack Coefficient
0.000231; !- Wind Coefficient

!- ===== ALL OBJECTS IN CLASS: ZONEVENTILATION:DESIGNFLOWRATE
=====

ZoneVentilation:DesignFlowRate,
 Ventilation_unit1, !- Name
 living_unit1, !- Zone or ZoneList Name
 always_avail, !- Schedule Name
 Flow/Zone, !- Design Flow Rate Calculation Method
 0.028336639274582, !- Design Flow Rate {m3/s}
 , !- Flow Rate per Zone Floor Area {m3/s-m2}
 , !- Flow Rate per Person {m3/s-person}
 , !- Air Changes per Hour {1/hr}
 Exhaust, !- Ventilation Type
 0, !- Fan Pressure Rise {Pa}
 1, !- Fan Total Efficiency
 1, !- Constant Term Coefficient
 0, !- Temperature Term Coefficient
 0, !- Velocity Term Coefficient
 0, !- Velocity Squared Term Coefficient
 -100, !- Minimum Indoor Temperature {C}
 , !- Minimum Indoor Temperature Schedule Name
 100, !- Maximum Indoor Temperature {C}
 , !- Maximum Indoor Temperature Schedule Name
 -100, !- Delta Temperature {deltaC}
 , !- Delta Temperature Schedule Name
 -100, !- Minimum Outdoor Temperature {C}
 , !- Minimum Outdoor Temperature Schedule Name
 100, !- Maximum Outdoor Temperature {C}
 , !- Maximum Outdoor Temperature Schedule Name
 40; !- Maximum Wind Speed {m/s}

!- ===== ALL OBJECTS IN CLASS: EXTERIOR:LIGHTS =====

Exterior:Lights,
 Exterior-Lights, !- Name
 ExteriorLightingProfile, !- Schedule Name
 58, !- Design Level {W}
 , !- Control Option
 Exterior-Lights; !- End-Use Subcategory

Exterior:Lights,
 Garage-Lights, !- Name
 LightingProfile_EELighting, !- Schedule Name

9.5, !- Design Level {W}
 , !- Control Option
 Exterior-Lights; !- End-Use Subcategory

!- ===== ALL OBJECTS IN CLASS: DESIGNSPECIFICATION:OUTDOORAIR
 =====

DesignSpecification:OutdoorAir,
 SZ DSOA living_unit1, !- Name
 Flow/Zone, !- Outdoor Air Method
 0, !- Outdoor Air Flow per Person {m3/s-person}
 , !- Outdoor Air Flow per Zone Floor Area {m3/s-m2}
 0; !- Outdoor Air Flow per Zone {m3/s}

!- ===== ALL OBJECTS IN CLASS: SIZING:ZONE =====

Sizing:Zone,
 living_unit1, !- Zone or ZoneList Name
 SupplyAirTemperature, !- Zone Cooling Design Supply Air Temperature Input Method
 12, !- Zone Cooling Design Supply Air Temperature {C}
 , !- Zone Cooling Design Supply Air Temperature Difference {deltaC}
 SupplyAirTemperature, !- Zone Heating Design Supply Air Temperature Input Method
 50, !- Zone Heating Design Supply Air Temperature {C}
 , !- Zone Heating Design Supply Air Temperature Difference {deltaC}
 0.008, !- Zone Cooling Design Supply Air Humidity Ratio {kgWater/kgDryAir}
 0.008, !- Zone Heating Design Supply Air Humidity Ratio {kgWater/kgDryAir}
 SZ DSOA living_unit1, !- Design Specification Outdoor Air Object Name
 , !- Zone Heating Sizing Factor
 , !- Zone Cooling Sizing Factor
 DesignDay, !- Cooling Design Air Flow Method
 , !- Cooling Design Air Flow Rate {m3/s}
 0.000762, !- Cooling Minimum Air Flow per Zone Floor Area {m3/s-m2}
 , !- Cooling Minimum Air Flow {m3/s}
 , !- Cooling Minimum Air Flow Fraction
 DesignDay, !- Heating Design Air Flow Method
 , !- Heating Design Air Flow Rate {m3/s}
 0.002032, !- Heating Maximum Air Flow per Zone Floor Area {m3/s-m2}
 0.1415762, !- Heating Maximum Air Flow {m3/s}
 0.3; !- Heating Maximum Air Flow Fraction

!- ===== ALL OBJECTS IN CLASS: SIZING:SYSTEM =====

Sizing:System,
 Central System_unit1, !- AirLoop Name
 Sensible, !- Type of Load to Size On
 autosize, !- Design Outdoor Air Flow Rate {m3/s}

1, !- Central Heating Maximum System Air Flow Ratio
 7, !- Preheat Design Temperature {C}
 0.008, !- Preheat Design Humidity Ratio {kgWater/kgDryAir}
 11, !- Precool Design Temperature {C}
 0.008, !- Precool Design Humidity Ratio {kgWater/kgDryAir}
 12, !- Central Cooling Design Supply Air Temperature {C}
 50, !- Central Heating Design Supply Air Temperature {C}
 NonCoincident, !- Type of Zone Sum to Use
 No, !- 100% Outdoor Air in Cooling
 No, !- 100% Outdoor Air in Heating
 0.008, !- Central Cooling Design Supply Air Humidity Ratio {kgWater/kgDryAir}
 0.008, !- Central Heating Design Supply Air Humidity Ratio {kgWater/kgDryAir}
 DesignDay, !- Cooling Supply Air Flow Rate Method
 , !- Cooling Supply Air Flow Rate {m3/s}
 , !- Cooling Supply Air Flow Rate Per Floor Area {m3/s-m2}
 , !- Cooling Fraction of Autosized Cooling Supply Air Flow Rate
 , !- Cooling Supply Air Flow Rate Per Unit Cooling Capacity {m3/s-W}
 DesignDay, !- Heating Supply Air Flow Rate Method
 , !- Heating Supply Air Flow Rate {m3/s}
 , !- Heating Supply Air Flow Rate Per Floor Area {m3/s-m2}
 , !- Heating Fraction of Autosized Heating Supply Air Flow Rate
 , !- Heating Fraction of Autosized Cooling Supply Air Flow Rate
 , !- Heating Supply Air Flow Rate Per Unit Heating Capacity {m3/s-W}
 ZoneSum, !- System Outdoor Air Method
 1.0, !- Zone Maximum Outdoor Air Fraction {dimensionless}
 CoolingDesignCapacity, !- Cooling Design Capacity Method
 autosize, !- Cooling Design Capacity {W}
 , !- Cooling Design Capacity Per Floor Area {W/m2}
 , !- Fraction of Autosized Cooling Design Capacity
 HeatingDesignCapacity, !- Heating Design Capacity Method
 autosize, !- Heating Design Capacity {W}
 , !- Heating Design Capacity Per Floor Area {W/m2}
 , !- Fraction of Autosized Heating Design Capacity
 OnOff; !- Central Cooling Capacity Control Method

!- ===== ALL OBJECTS IN CLASS: SIZING:PLANT =====

Sizing:Plant,
 DHW Loop_unit1, !- Plant or Condenser Loop Name
 Heating, !- Loop Type
 48.8888888888889, !- Design Loop Exit Temperature {C}
 5.55555555555556; !- Loop Design Temperature Difference {deltaC}

!- ===== ALL OBJECTS IN CLASS: ZONECONTROL:THERMOSTAT =====

ZoneControl:Thermostat,
 Zone Thermostat_unit1, !- Name

```

living_unit1,      !- Zone or ZoneList Name
zone_control_type, !- Control Type Schedule Name
ThermostatSetpoint:DualSetpoint, !- Control 1 Object Type
thermostat_living Dual SP Control; !- Control 1 Name

!- ===== ALL OBJECTS IN CLASS: THERMOSTATSETPOINT:DUALSETPOINT
=====

ThermostatSetpoint:DualSetpoint,
  thermostat_living Dual SP Control, !- Name
  myStartHeating,      !- Heating Setpoint Temperature Schedule Name
  myStartCooling;      !- Cooling Setpoint Temperature Schedule Name

!- ===== ALL OBJECTS IN CLASS: AIRTERMINAL:SINGLEDUCT:UNCONTROLLED
=====

AirTerminal:SingleDuct:Uncontrolled,
  ZoneDirectAir_unit1,  !- Name
  always_avail,         !- Availability Schedule Name
  Zone Inlet Node_unit1, !- Zone Supply Air Node Name
  autosize;             !- Maximum Air Flow Rate {m3/s}

!- ===== ALL OBJECTS IN CLASS: ZONEHVAC:EQUIPMENTLIST =====

ZoneHVAC:EquipmentList,
  ZONEEQUIPMENT_unit1,  !- Name
  AirTerminal:SingleDuct:Uncontrolled, !- Zone Equipment 1 Object Type
  ZoneDirectAir_unit1,  !- Zone Equipment 1 Name
  1,                    !- Zone Equipment 1 Cooling Sequence
  1;                    !- Zone Equipment 1 Heating or No-Load Sequence

!- ===== ALL OBJECTS IN CLASS: ZONEHVAC:EQUIPMENTCONNECTIONS
=====

ZoneHVAC:EquipmentConnections,
  living_unit1,      !- Zone Name
  ZoneEquipment_unit1, !- Zone Conditioning Equipment List Name
  zone inlet nodes_unit1, !- Zone Air Inlet Node or NodeList Name
  ,                  !- Zone Air Exhaust Node or NodeList Name
  Zone Node_unit1,    !- Zone Air Node Name
  Zone Outlet Node_unit1; !- Zone Return Air Node Name

!- ===== ALL OBJECTS IN CLASS: FAN:ONOFF =====

```

```

Fan:OnOff,
  Supply Fan_unit1,      !- Name
  always_avail,          !- Availability Schedule Name
  0.377,                  !- Fan Total Efficiency
  400,                    !- Pressure Rise {Pa}
  autosize,               !- Maximum Flow Rate {m3/s}
  0.65,                   !- Motor Efficiency
  1,                      !- Motor In Airstream Fraction
  air loop inlet node_unit1, !- Air Inlet Node Name
  cooling coil air inlet node_unit1, !- Air Outlet Node Name
  ,                       !- Fan Power Ratio Function of Speed Ratio Curve Name
  ,                       !- Fan Efficiency Ratio Function of Speed Ratio Curve Name
  General;                !- End-Use Subcategory

```

!- ===== ALL OBJECTS IN CLASS: COIL:COOLING:DX:SINGLESPEED =====

```

Coil:Cooling:DX:SingleSpeed,
  DX Cooling Coil_unit1, !- Name
  always_avail,          !- Availability Schedule Name
  5275.279799999999,      !- Gross Rated Total Cooling Capacity {W}
  autosize,               !- Gross Rated Sensible Heat Ratio
  3.97008850025305,       !- Gross Rated Cooling COP {W/W}
  autosize,               !- Rated Air Flow Rate {m3/s}
  ,                       !- Rated Evaporator Fan Power Per Volume Flow Rate {W/(m3/s)}
  Cooling Coil Air Inlet Node_unit1, !- Air Inlet Node Name
  Heating Coil Air Inlet Node_unit1, !- Air Outlet Node Name
  Cool-Cap-FT,            !- Total Cooling Capacity Function of Temperature Curve Name
  ConstantCubic,          !- Total Cooling Capacity Function of Flow Fraction Curve Name
  Cool-EIR-FT,            !- Energy Input Ratio Function of Temperature Curve Name
  ConstantCubic,          !- Energy Input Ratio Function of Flow Fraction Curve Name
  Cool-PLF-fPLR;          !- Part Load Fraction Correlation Curve Name

```

!- ===== ALL OBJECTS IN CLASS: COIL:HEATING:FUEL =====

```

Coil:Heating:Fuel,
  Main gas heating coil_unit1, !- Name
  always_avail,               !- Availability Schedule Name
  NaturalGas,                 !- Fuel Type
  0.78,                       !- Burner Efficiency
  1465.3555,                  !- Nominal Capacity {W}
  heating coil air inlet node_unit1, !- Air Inlet Node Name
  air loop outlet node_unit1; !- Air Outlet Node Name

```

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC:UNITARYHEATCOOL =====

```

AirLoopHVAC:UnitaryHeatCool,

```

ACandF_unit1, !- Name
 always_avail, !- Availability Schedule Name
 air loop inlet node_unit1, !- Unitary System Air Inlet Node Name
 air loop outlet node_unit1, !- Unitary System Air Outlet Node Name
 fan_cycle, !- Supply Air Fan Operating Mode Schedule Name
 80, !- Maximum Supply Air Temperature {C}
 autosize, !- Cooling Supply Air Flow Rate {m3/s}
 autosize, !- Heating Supply Air Flow Rate {m3/s}
 0, !- No Load Supply Air Flow Rate {m3/s}
 living_unit1, !- Controlling Zone or Thermostat Location
 Fan:OnOff, !- Supply Fan Object Type
 Supply Fan_unit1, !- Supply Fan Name
 BlowThrough, !- Fan Placement
 Coil:Heating:Fuel, !- Heating Coil Object Type
 Main gas Heating Coil_unit1, !- Heating Coil Name
 Coil:Cooling:DX:SingleSpeed, !- Cooling Coil Object Type
 DX Cooling Coil_unit1, !- Cooling Coil Name
 None; !- Dehumidification Control Type

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC =====

AirLoopHVAC,
 Central System_unit1, !- Name
 , !- Controller List Name
 availability list, !- Availability Manager List Name
 autosize, !- Design Supply Air Flow Rate {m3/s}
 Air Loop Branches_unit1, !- Branch List Name
 , !- Connector List Name
 Air Loop Inlet Node_unit1, !- Supply Side Inlet Node Name
 Return Air Mixer Outlet_unit1, !- Demand Side Outlet Node Name
 Zone Equipment Inlet Node_unit1, !- Demand Side Inlet Node Names
 Air Loop Outlet Node_unit1; !- Supply Side Outlet Node Names

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC:ZONESPLITTER =====

AirLoopHVAC:ZoneSplitter,
 Zone Supply Air Splitter_unit1, !- Name
 Zone Equipment Inlet Node_unit1, !- Inlet Node Name
 Zone Inlet Node_unit1; !- Outlet 1 Node Name

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC:SUPPLYPATH =====

AirLoopHVAC:SupplyPath,
 SupplyPath_unit1, !- Name
 Zone Equipment Inlet Node_unit1, !- Supply Air Path Inlet Node Name
 AirLoopHVAC:ZoneSplitter,!- Component 1 Object Type

Zone Supply Air Splitter_unit1; !- Component 1 Name

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC:ZONEMIXER =====

AirLoopHVAC:ZoneMixer,
Zone Return Air Mixer_unit1, !- Name
Return Air Mixer Outlet_unit1, !- Outlet Node Name
Zone Outlet Node_unit1; !- Inlet 1 Node Name

!- ===== ALL OBJECTS IN CLASS: AIRLOOPHVAC:RETURNPATH =====

AirLoopHVAC:ReturnPath,
ReturnPath_unit1, !- Name
Return Air Mixer Outlet_unit1, !- Return Air Path Outlet Node Name
AirLoopHVAC:ZoneMixer, !- Component 1 Object Type
Zone Return Air Mixer_unit1; !- Component 1 Name

!- ===== ALL OBJECTS IN CLASS: BRANCH =====

Branch,
Air Loop Main Branch_unit1, !- Name
, !- Pressure Drop Curve Name
AirLoopHVAC:UnitaryHeatCool, !- Component 1 Object Type
ACandF_unit1, !- Component 1 Name
Air Loop Inlet Node_unit1, !- Component 1 Inlet Node Name
Air loop outlet node_unit1; !- Component 1 Outlet Node Name

Branch,
Mains Inlet Branch_unit1, !- Name
, !- Pressure Drop Curve Name
Pump:VariableSpeed, !- Component 1 Object Type
Mains Pressure_unit1, !- Component 1 Name
Mains Inlet Node_unit1, !- Component 1 Inlet Node Name
Mains Pressure Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
Water Heater Branch_unit1, !- Name
, !- Pressure Drop Curve Name
WaterHeater:Mixed, !- Component 1 Object Type
Water Heater_unit1, !- Component 1 Name
Water Heater Use Inlet Node_unit1, !- Component 1 Inlet Node Name
Water Heater Use Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
DHW Supply Outlet Branch_unit1, !- Name
, !- Pressure Drop Curve Name

Pipe:Adiabatic, !- Component 1 Object Type
DHW Supply Outlet Pipe_unit1, !- Component 1 Name
DHW Supply Outlet Pipe Inlet Node_unit1, !- Component 1 Inlet Node Name
DHW Supply Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 DHW Demand Inlet Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 Pipe:Adiabatic, !- Component 1 Object Type
 DHW Demand Inlet Pipe_unit1, !- Component 1 Name
 DHW Demand Inlet Node_unit1, !- Component 1 Inlet Node Name
 DHW Demand Inlet Pipe Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Water Sink Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 WaterUse:Connections, !- Component 1 Object Type
 DHW Sinks_unit1, !- Component 1 Name
 Water Sink Inlet Node_unit1, !- Component 1 Inlet Node Name
 Water Sink outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Water Shower Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 WaterUse:Connections, !- Component 1 Object Type
 DHW Showers_unit1, !- Component 1 Name
 Water Shower Inlet Node_unit1, !- Component 1 Inlet Node Name
 Water Shower Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Water ClothesWasher Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 WaterUse:Connections, !- Component 1 Object Type
 DHW ClothesWasher_unit1, !- Component 1 Name
 Water ClothesWasher Inlet Node_unit1, !- Component 1 Inlet Node Name
 Water ClothesWasher Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Water Dishwasher Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 WaterUse:Connections, !- Component 1 Object Type
 DHW DishWasher_unit1, !- Component 1 Name
 Water DishWasher Inlet Node_unit1, !- Component 1 Inlet Node Name
 Water DishWasher outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Water Bath Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 WaterUse:Connections, !- Component 1 Object Type

DHW Baths_unit1, !- Component 1 Name
Water Bath Inlet Node_unit1, !- Component 1 Inlet Node Name
Water bath Outlet Node_unit1; !- Component 1 Outlet Node Name

Branch,
 Mains Makeup Branch_unit1, !- Name
 , !- Pressure Drop Curve Name
 Pipe:Adiabatic, !- Component 1 Object Type
 Mains Makeup Pipe_unit1, !- Component 1 Name
 Mains Makeup Pipe Inlet Node_unit1, !- Component 1 Inlet Node Name
 Mains Makeup Node_unit1; !- Component 1 Outlet Node Name

!- ===== ALL OBJECTS IN CLASS: BRANCHLIST =====

BranchList,
 Air Loop Branches_unit1, !- Name
 Air Loop Main Branch_unit1; !- Branch 1 Name

BranchList,
 DHW Supply Branches_unit1, !- Name
 Mains Inlet Branch_unit1,!- Branch 1 Name
 Water Heater Branch_unit1, !- Branch 2 Name
 DHW Supply Outlet Branch_unit1; !- Branch 3 Name

BranchList,
 DHW Demand Branches_unit1, !- Name
 DHW Demand Inlet Branch_unit1, !- Branch 1 Name
 Water Sink Branch_unit1, !- Branch 2 Name
 Water Shower Branch_unit1, !- Branch 3 Name
 Water ClothesWasher Branch_unit1, !- Branch 4 Name
 Water Dishwasher Branch_unit1, !- Branch 5 Name
 Water Bath Branch_unit1, !- Branch 6 Name
 Mains Makeup Branch_unit1; !- Branch 7 Name

!- ===== ALL OBJECTS IN CLASS: CONNECTOR:SPLITTER =====

Connector:Splitter,
 DHW Demand Splitter_unit1, !- Name
 DHW Demand Inlet Branch_unit1, !- Inlet Branch Name
 Water Sink Branch_unit1, !- Outlet Branch 1 Name
 Water Shower Branch_unit1, !- Outlet Branch 2 Name
 Water ClothesWasher Branch_unit1, !- Outlet Branch 3 Name
 Water Dishwasher Branch_unit1, !- Outlet Branch 4 Name
 Water Bath Branch_unit1; !- Outlet Branch 5 Name

Connector:Splitter,
 DHW Supply Splitter_unit1, !- Name

Mains Inlet Branch_unit1, !- Inlet Branch Name
Water Heater Branch_unit1; !- Outlet Branch 1 Name

!- ===== ALL OBJECTS IN CLASS: CONNECTOR:MIXER =====

Connector:Mixer,
DHW Demand Mixer_unit1, !- Name
Mains Makeup Branch_unit1, !- Outlet Branch Name
Water Sink Branch_unit1, !- Inlet Branch 1 Name
Water Shower Branch_unit1, !- Inlet Branch 2 Name
Water ClothesWasher Branch_unit1, !- Inlet Branch 3 Name
Water Dishwasher Branch_unit1, !- Inlet Branch 4 Name
Water Bath Branch_unit1; !- Inlet Branch 5 Name

Connector:Mixer,
DHW Supply Mixer_unit1, !- Name
DHW Supply Outlet Branch_unit1, !- Outlet Branch Name
Water Heater Branch_unit1; !- Inlet Branch 1 Name

!- ===== ALL OBJECTS IN CLASS: CONNECTORLIST =====

ConnectorList,
DHW Demand Connectors_unit1, !- Name
Connector:Splitter, !- Connector 1 Object Type
DHW Demand Splitter_unit1, !- Connector 1 Name
Connector:Mixer, !- Connector 2 Object Type
DHW Demand Mixer_unit1; !- Connector 2 Name

ConnectorList,
DHW Supply Connectors_unit1, !- Name
Connector:Splitter, !- Connector 1 Object Type
DHW Supply Splitter_unit1, !- Connector 1 Name
Connector:Mixer, !- Connector 2 Object Type
DHW Supply Mixer_unit1; !- Connector 2 Name

!- ===== ALL OBJECTS IN CLASS: NODELIST =====

NodeList,
Zone Inlet Nodes_unit1, !- Name
Zone Inlet Node_unit1; !- Node 1 Name

!- ===== ALL OBJECTS IN CLASS: OUTDOORAIR:NODE =====

OutdoorAir:Node,
outside air inlet node_unit1, !- Name

```

0.914355407629293;    !- Height Above Ground {m}

OutdoorAir:Node,
  outdoor air node_unit1, !- Name
  1;                      !- Height Above Ground {m}

!- ===== ALL OBJECTS IN CLASS: PIPE:ADIABATIC =====

Pipe:Adiabatic,
  DHW Supply Outlet Pipe_unit1, !- Name
  DHW Supply Outlet Pipe Inlet Node_unit1, !- Inlet Node Name
  DHW Supply Outlet Node_unit1; !- Outlet Node Name

Pipe:Adiabatic,
  Mains Makeup Pipe_unit1, !- Name
  Mains Makeup Pipe Inlet Node_unit1, !- Inlet Node Name
  Mains Makeup Node_unit1; !- Outlet Node Name

Pipe:Adiabatic,
  DHW Demand Inlet Pipe_unit1, !- Name
  DHW Demand Inlet Node_unit1, !- Inlet Node Name
  DHW Demand Inlet Pipe Outlet Node_unit1; !- Outlet Node Name

!- ===== ALL OBJECTS IN CLASS: PUMP:VARIABLESPEED =====

Pump:VariableSpeed,
  Mains Pressure_unit1, !- Name
  Mains Inlet Node_unit1, !- Inlet Node Name
  Mains Pressure Outlet Node_unit1, !- Outlet Node Name
  autosize,             !- Design Maximum Flow Rate {m3/s}
  1,                    !- Design Pump Head {Pa}
  0,                    !- Design Power Consumption {W}
  1,                    !- Motor Efficiency
  0,                    !- Fraction of Motor Inefficiencies to Fluid Stream
  0,                    !- Coefficient 1 of the Part Load Performance Curve
  1,                    !- Coefficient 2 of the Part Load Performance Curve
  0,                    !- Coefficient 3 of the Part Load Performance Curve
  0,                    !- Coefficient 4 of the Part Load Performance Curve
  0,                    !- Design Minimum Flow Rate {m3/s}
  Intermittent;         !- Pump Control Type

!- ===== ALL OBJECTS IN CLASS: WATERHEATER:MIXED =====

WaterHeater:Mixed,
  Water Heater_unit1, !- Name
  0.15141644,         !- Tank Volume {m3}

```

```

dhw_setpt,      !- Setpoint Temperature Schedule Name
2,              !- Deadband Temperature Difference {deltaC}
50,             !- Maximum Temperature Limit {C}
Cycle,          !- Heater Control Type
autosize,       !- Heater Maximum Capacity {W}
0,              !- Heater Minimum Capacity {W}
0,              !- Heater Ignition Minimum Flow Rate {m3/s}
,              !- Heater Ignition Delay {s}
naturalgas,     !- Heater Fuel Type
0.8,            !- Heater Thermal Efficiency
,              !- Part Load Factor Curve Name
,              !- Off Cycle Parasitic Fuel Consumption Rate {W}
,              !- Off Cycle Parasitic Fuel Type
,              !- Off Cycle Parasitic Heat Fraction to Tank
,              !- On Cycle Parasitic Fuel Consumption Rate {W}
,              !- On Cycle Parasitic Fuel Type
,              !- On Cycle Parasitic Heat Fraction to Tank
Zone,           !- Ambient Temperature Indicator
,              !- Ambient Temperature Schedule Name
living_unit1,   !- Ambient Temperature Zone Name
,              !- Ambient Temperature Outdoor Air Node Name
5.71166831384137, !- Off Cycle Loss Coefficient to Ambient Temperature {W/K}
1,              !- Off Cycle Loss Fraction to Zone
5.71166831384137, !- On Cycle Loss Coefficient to Ambient Temperature {W/K}
1,              !- On Cycle Loss Fraction to Zone
0,              !- Peak Use Flow Rate {m3/s}
,              !- Use Flow Rate Fraction Schedule Name
,              !- Cold Water Supply Temperature Schedule Name
Water Heater use inlet node_unit1, !- Use Side Inlet Node Name
Water Heater use outlet node_unit1, !- Use Side Outlet Node Name
1,              !- Use Side Effectiveness
,              !- Source Side Inlet Node Name
,              !- Source Side Outlet Node Name
1,              !- Source Side Effectiveness
autosize,       !- Use Side Design Flow Rate {m3/s}
0,              !- Source Side Design Flow Rate {m3/s}
1.5;            !- Indirect Water Heating Recovery Time {hr}

```

!- ===== ALL OBJECTS IN CLASS: WATERHEATER:SIZING =====

```

WaterHeater:Sizing,
  Water Heater_unit1,    !- WaterHeater Name
  ResidentialHUD-FHMinimum, !- Design Mode
  ,                      !- Time Storage Can Meet Peak Draw {hr}
  ,                      !- Time for Tank Recovery {hr}
  ,                      !- Nominal Tank Volume for Autosizing Plant Connections {m3}
  3,                    !- Number of Bedrooms
  3;                    !- Number of Bathrooms

```

!- ===== ALL OBJECTS IN CLASS: PLANTLOOP =====

```
PlantLoop,
  DHW Loop_unit1,      !- Name
  Water,               !- Fluid Type
  ,                   !- User Defined Fluid Type
  DHW Loop Operation_unit1,!- Plant Equipment Operation Scheme Name
  DHW Supply Outlet Node_unit1, !- Loop Temperature Setpoint Node Name
  100,                 !- Maximum Loop Temperature {C}
  0,                   !- Minimum Loop Temperature {C}
  autosize,            !- Maximum Loop Flow Rate {m3/s}
  0,                   !- Minimum Loop Flow Rate {m3/s}
  autocalculate,       !- Plant Loop Volume {m3}
  Mains Inlet Node_unit1, !- Plant Side Inlet Node Name
  DHW Supply Outlet Node_unit1, !- Plant Side Outlet Node Name
  DHW Supply Branches_unit1, !- Plant Side Branch List Name
  DHW Supply Connectors_unit1, !- Plant Side Connector List Name
  DHW Demand Inlet Node_unit1, !- Demand Side Inlet Node Name
  Mains Makeup Node_unit1, !- Demand Side Outlet Node Name
  DHW Demand Branches_unit1, !- Demand Side Branch List Name
  DHW Demand Connectors_unit1, !- Demand Side Connector List Name
  Optimal;             !- Load Distribution Scheme
```

!- ===== ALL OBJECTS IN CLASS: PLANTEQUIPMENTLIST =====

```
PlantEquipmentList,
  DHW Plant Equipment_unit1, !- Name
  WaterHeater:Mixed,        !- Equipment 1 Object Type
  Water Heater_unit1;       !- Equipment 1 Name
```

!- ===== ALL OBJECTS IN CLASS: PLANTEQUIPMENTOPERATION:HEATINGLOAD =====

```
PlantEquipmentOperation:HeatingLoad,
  DHW Control Scheme_unit1,!- Name
  0.0,                       !- Load Range 1 Lower Limit {W}
  10000000000000000,        !- Load Range 1 Upper Limit {W}
  DHW Plant Equipment_unit1; !- Range 1 Equipment List Name
```

!- ===== ALL OBJECTS IN CLASS: PLANTEQUIPMENTOPERATIONSCHEMES =====

```
PlantEquipmentOperationSchemes,
  DHW Loop Operation_unit1,!- Name
```

PlantEquipmentOperation:HeatingLoad, !- Control Scheme 1 Object Type
 DHW Control Scheme_unit1,!- Control Scheme 1 Name
 always_avail; !- Control Scheme 1 Schedule Name

!- ===== ALL OBJECTS IN CLASS: EXTERNALINTERFACE =====

ExternalInterface,
 FunctionalMockupUnitImport; !- Name of External Interface

!- ===== ALL OBJECTS IN CLASS:
 EXTERNALINTERFACE:FUNCTIONALMOCKUPUNITIMPORT =====

ExternalInterface:FunctionalMockupUnitImport,
 Joe_ep_fmu.fmu, !- FMU File Name
 0, !- FMU Timeout {ms}
 1; !- FMU LoggingOn

!- ===== ALL OBJECTS IN CLASS:
 EXTERNALINTERFACE:FUNCTIONALMOCKUPUNITIMPORT:FROM:VARIABLE =====

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
 living_unit1, !- Output:Variable Index Key Name
 Zone Air Temperature, !- Output:Variable Name
 Joe_ep_fmu.fmu, !- FMU File Name
 Joe_ep_fmu, !- FMU Instance Name
 epSendZoneMeanAirTemp; !- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
 Environment, !- Output:Variable Index Key Name
 Site Outdoor Air Drybulb Temperature, !- Output:Variable Name
 Joe_ep_fmu.fmu, !- FMU File Name
 Joe_ep_fmu, !- FMU Instance Name
 epSendOutdoorAirTemp; !- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
 living_unit1, !- Output:Variable Index Key Name
 Zone Air Relative Humidity, !- Output:Variable Name
 Joe_ep_fmu.fmu, !- FMU File Name
 Joe_ep_fmu, !- FMU Instance Name
 epSendZoneHumidity; !- FMU Variable Name

!- ===== ALL OBJECTS IN CLASS:
 EXTERNALINTERFACE:FUNCTIONALMOCKUPUNITIMPORT:TO:SCHEDULE =====

ExternalInterface:FunctionalMockupUnitImport:To:Schedule,


```

myStartHeating,      !- Name
myTemps,             !- Schedule Type Limits Names
Joe_ep_fmu.fmu,      !- FMU File Name
Joe_ep_fmu,          !- FMU Instance Name
epGetStartHeating,   !- FMU Variable Name
18;                  !- Initial Value

ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
myStartCooling,      !- Name
myTemps,             !- Schedule Type Limits Names
Joe_ep_fmu.fmu,      !- FMU File Name
Joe_ep_fmu,          !- FMU Instance Name
epGetStartCooling,   !- FMU Variable Name
26;                  !- Initial Value

!- ===== ALL OBJECTS IN CLASS: AVAILABILITYMANAGER:SCHEDULED =====

AvailabilityManager:Scheduled,
  System availability, !- Name
  always_avail;       !- Schedule Name

!- ===== ALL OBJECTS IN CLASS: AVAILABILITYMANAGERASSIGNMENTLIST =====

AvailabilityManagerAssignmentList,
  availability list,   !- Name
  AvailabilityManager:Scheduled, !- Availability Manager 1 Object Type
  System availability; !- Availability Manager 1 Name

!- ===== ALL OBJECTS IN CLASS: SETPOINTMANAGER:SCHEDULED =====

SetpointManager:Scheduled,
  DHW Loop Setpoint Manager_unit1, !- Name
  Temperature,                    !- Control Variable
  DHWSupplySetpoint,              !- Schedule Name
  DHW Supply Outlet Node_unit1; !- Setpoint Node or NodeList Name

!- ===== ALL OBJECTS IN CLASS: WATERUSE:EQUIPMENT =====

WaterUse:Equipment,
  Clothes Washer_unit1, !- Name
  Domestic Hot Water,   !- End-Use Subcategory
  1.6219189818e-06,     !- Peak Flow Rate {m3/s}
  ClothesWasher,        !- Flow Rate Fraction Schedule Name
  CWWaterTempSchedule;  !- Target Temperature Schedule Name

```

```

WaterUse:Equipment,
  Dishwasher_unit1,      !- Name
  Domestic Hot Water,    !- End-Use Subcategory
  6.36685353e-07,        !- Peak Flow Rate {m3/s}
  Dishwasher,            !- Flow Rate Fraction Schedule Name
  DWWaterTempSchedule;   !- Target Temperature Schedule Name

```

```

WaterUse:Equipment,
  Sinks_unit1,           !- Name
  Domestic Hot Water,    !- End-Use Subcategory
  2.1244858873e-06,      !- Peak Flow Rate {m3/s}
  Sinks,                  !- Flow Rate Fraction Schedule Name
  SSBWaterTempSchedule;  !- Target Temperature Schedule Name

```

```

WaterUse:Equipment,
  Showers_unit1,         !- Name
  Domestic Hot Water,    !- End-Use Subcategory
  3.7816251714e-06,      !- Peak Flow Rate {m3/s}
  Showers,                !- Flow Rate Fraction Schedule Name
  SSBWaterTempSchedule;  !- Target Temperature Schedule Name

```

```

WaterUse:Equipment,
  Baths_unit1,           !- Name
  Domestic Hot Water,    !- End-Use Subcategory
  9.695682173e-07,       !- Peak Flow Rate {m3/s}
  Baths,                  !- Flow Rate Fraction Schedule Name
  SSBWaterTempSchedule;  !- Target Temperature Schedule Name

```

!- ===== ALL OBJECTS IN CLASS: WATERUSE:CONNECTIONS =====

```

WaterUse:Connections,
  DHW Sinks_unit1,       !- Name
  Water Sink Inlet Node_unit1, !- Inlet Node Name
  Water Sink Outlet Node_unit1, !- Outlet Node Name
  ,                       !- Supply Water Storage Tank Name
  ,                       !- Reclamation Water Storage Tank Name
  ,                       !- Hot Water Supply Temperature Schedule Name
  ,                       !- Cold Water Supply Temperature Schedule Name
  None,                  !- Drain Water Heat Exchanger Type
  ,                       !- Drain Water Heat Exchanger Destination
  ,                       !- Drain Water Heat Exchanger U-Factor Times Area {W/K}
  Sinks_unit1;           !- Water Use Equipment 1 Name

```

```

WaterUse:Connections,
  DHW Showers_unit1,     !- Name
  Water Shower Inlet Node_unit1, !- Inlet Node Name
  Water Shower Outlet Node_unit1, !- Outlet Node Name

```

, !- Supply Water Storage Tank Name
 , !- Reclamation Water Storage Tank Name
 , !- Hot Water Supply Temperature Schedule Name
 , !- Cold Water Supply Temperature Schedule Name
 None, !- Drain Water Heat Exchanger Type
 , !- Drain Water Heat Exchanger Destination
 , !- Drain Water Heat Exchanger U-Factor Times Area {W/K}
 Showers_unit1; !- Water Use Equipment 1 Name

WaterUse:Connections,

DHW ClothesWasher_unit1, !- Name
 Water ClothesWasher Inlet Node_unit1, !- Inlet Node Name
 Water ClothesWasher Outlet Node_unit1, !- Outlet Node Name
 , !- Supply Water Storage Tank Name
 , !- Reclamation Water Storage Tank Name
 , !- Hot Water Supply Temperature Schedule Name
 , !- Cold Water Supply Temperature Schedule Name
 None, !- Drain Water Heat Exchanger Type
 , !- Drain Water Heat Exchanger Destination
 , !- Drain Water Heat Exchanger U-Factor Times Area {W/K}
 Clothes Washer_unit1; !- Water Use Equipment 1 Name

WaterUse:Connections,

DHW DishWasher_unit1, !- Name
 Water DishWasher Inlet Node_unit1, !- Inlet Node Name
 Water DishWasher Outlet Node_unit1, !- Outlet Node Name
 , !- Supply Water Storage Tank Name
 , !- Reclamation Water Storage Tank Name
 , !- Hot Water Supply Temperature Schedule Name
 , !- Cold Water Supply Temperature Schedule Name
 None, !- Drain Water Heat Exchanger Type
 , !- Drain Water Heat Exchanger Destination
 , !- Drain Water Heat Exchanger U-Factor Times Area {W/K}
 Dishwasher_unit1; !- Water Use Equipment 1 Name

WaterUse:Connections,

DHW Baths_unit1, !- Name
 Water Bath Inlet Node_unit1, !- Inlet Node Name
 Water Bath Outlet Node_unit1, !- Outlet Node Name
 , !- Supply Water Storage Tank Name
 , !- Reclamation Water Storage Tank Name
 , !- Hot Water Supply Temperature Schedule Name
 , !- Cold Water Supply Temperature Schedule Name
 None, !- Drain Water Heat Exchanger Type
 , !- Drain Water Heat Exchanger Destination
 , !- Drain Water Heat Exchanger U-Factor Times Area {W/K}
 Baths_unit1; !- Water Use Equipment 1 Name

!- ===== ALL OBJECTS IN CLASS: CURVE:QUADRATIC =====

```
Curve:Quadratic,
  HPACCoolCapFFF,      !- Name
  0.8,                  !- Coefficient1 Constant
  0.2,                  !- Coefficient2 x
  0.0,                  !- Coefficient3 x**2
  0.5,                  !- Minimum Value of x
  1.5;                  !- Maximum Value of x
```

```
Curve:Quadratic,
  HPACCOOLEIRFFF,      !- Name
  1.156,                !- Coefficient1 Constant
  -0.1816,              !- Coefficient2 x
  0.0256,               !- Coefficient3 x**2
  0.5,                  !- Minimum Value of x
  1.5;                  !- Maximum Value of x
```

```
Curve:Quadratic,
  HPACCOOLPLFFPLR,     !- Name
  0.85,                 !- Coefficient1 Constant
  0.15,                 !- Coefficient2 x
  0.0,                  !- Coefficient3 x**2
  0.0,                  !- Minimum Value of x
  1.0;                  !- Maximum Value of x
```

```
Curve:Quadratic,
  HPACHeatEIRFFF,      !- Name
  1.3824,               !- Coefficient1 Constant
  -0.4336,              !- Coefficient2 x
  0.0512,               !- Coefficient3 x**2
  0.0,                  !- Minimum Value of x
  1.0;                  !- Maximum Value of x
```

! NREL benchmark Cool-PLF-fPLR

```
Curve:Quadratic,
  Cool-PLF-fPLR,       !- Name
  0.80141423,          !- Coefficient1 Constant
  0.23744685,          !- Coefficient2 x
  -0.0393773,          !- Coefficient3 x**2
  0,                   !- Minimum Value of x
  1,                   !- Maximum Value of x
  0.7,                 !- Minimum Curve Output
  1;                   !- Maximum Curve Output
```

!- ===== ALL OBJECTS IN CLASS: CURVE:CUBIC =====

```
Curve:Cubic,
```

```

HPACHeatCapFT,      !- Name
0.758746,           !- Coefficient1 Constant
0.027626,           !- Coefficient2 x
0.000148716,        !- Coefficient3 x**2
0.0000034992,       !- Coefficient4 x**3
-20.0,              !- Minimum Value of x
20.0;               !- Maximum Value of x

Curve:Cubic,
  HPACHeatCapFFF,    !- Name
  0.84,              !- Coefficient1 Constant
  0.16,              !- Coefficient2 x
  0.0,               !- Coefficient3 x**2
  0.0,               !- Coefficient4 x**3
  0.5,               !- Minimum Value of x
  1.5;               !- Maximum Value of x

Curve:Cubic,
  HPACHeatEIRFT,     !- Name
  1.19248,           !- Coefficient1 Constant
  -0.0300438,        !- Coefficient2 x
  0.00103745,        !- Coefficient3 x**2
  -0.000023328,      !- Coefficient4 x**3
  -20.0,             !- Minimum Value of x
  20.0;              !- Maximum Value of x

Curve:Cubic,
  Fan-EIR-fPLR,      !- Name
  0.00000000,        !- Coefficient1 Constant
  1.00000000,        !- Coefficient2 x
  0.00000000,        !- Coefficient3 x**2
  0.00000000,        !- Coefficient4 x**3
  0,                  !- Minimum Value of x
  1,                  !- Maximum Value of x
  0,                  !- Minimum Curve Output
  1;                  !- Maximum Curve Output

Curve:Cubic,
  ConstantCubic,     !- Name
  1,                  !- Coefficient1 Constant
  0,                  !- Coefficient2 x
  0,                  !- Coefficient3 x**2
  0,                  !- Coefficient4 x**3
  -100,              !- Minimum Value of x
  100;               !- Maximum Value of x

!- ===== ALL OBJECTS IN CLASS: CURVE:BIQUADRATIC =====

```

Curve:Biquadratic,
 HPACCoolCapFT, !- Name
 0.766956, !- Coefficient1 Constant
 0.0107756, !- Coefficient2 x
 -0.0000414703, !- Coefficient3 x**2
 0.00134961, !- Coefficient4 y
 -0.000261144, !- Coefficient5 y**2
 0.000457488, !- Coefficient6 x*y
 12.77778, !- Minimum Value of x
 23.88889, !- Maximum Value of x
 21.11111, !- Minimum Value of y
 46.11111; !- Maximum Value of y

Curve:Biquadratic,
 HPACCOOLEIRFT, !- Name
 0.297145, !- Coefficient1 Constant
 0.0430933, !- Coefficient2 x
 -0.000748766, !- Coefficient3 x**2
 0.00597727, !- Coefficient4 y
 0.000482112, !- Coefficient5 y**2
 -0.000956448, !- Coefficient6 x*y
 12.77778, !- Minimum Value of x
 23.88889, !- Maximum Value of x
 21.11111, !- Minimum Value of y
 46.11111; !- Maximum Value of y

Curve:Biquadratic,
 Defrost_EIR_FT, !- Name
 1, !- Coefficient1 Constant
 0, !- Coefficient2 x
 0, !- Coefficient3 x**2
 0, !- Coefficient4 y
 0, !- Coefficient5 y**2
 0, !- Coefficient6 x*y
 0, !- Minimum Value of x
 100, !- Maximum Value of x
 0, !- Minimum Value of y
 100; !- Maximum Value of y

! NREL benchmark Cool-CAP-FT

Curve:Biquadratic,
 Cool-Cap-fT, !- Name
 1.26489391, !- Coefficient1 Constant
 -0.035054982, !- Coefficient2 x
 0.00211086, !- Coefficient3 x**2
 -0.001526886, !- Coefficient4 y
 -0.0000070308, !- Coefficient5 y**2
 -0.0004691844, !- Coefficient6 x*y
 -100, !- Minimum Value of x

100, !- Maximum Value of x
-100, !- Minimum Value of y
100; !- Maximum Value of y

! NREL benchmark Cool-EIR-FT

Curve:Biquadratic,

Cool-EIR-FT, !- Name
0.38402403, !- Coefficient1 Constant
0.029696724, !- Coefficient2 x
-0.0011329308, !- Coefficient3 x**2
0.006490674, !- Coefficient4 y
0.0002626992, !- Coefficient5 y**2
-0.0001207224, !- Coefficient6 x*y
-100, !- Minimum Value of x
100, !- Maximum Value of x
-100, !- Minimum Value of y
100; !- Maximum Value of y

!- ===== ALL OBJECTS IN CLASS: OUTPUT:VARIABLEDICTIONARY =====

Output:VariableDictionary,

Regular; !- Key Field

!- ===== ALL OBJECTS IN CLASS: OUTPUT:CONSTRUCTIONS =====

Output:Constructions,

Constructions, !- Details Type 1
Materials; !- Details Type 2

!- ===== ALL OBJECTS IN CLASS: OUTPUT:TABLE:SUMMARYREPORTS =====

Output:Table:SummaryReports,

InputVerificationandResultsSummary, !- Report 1 Name
EquipmentSummary, !- Report 2 Name
ClimaticDataSummary, !- Report 3 Name
EnvelopeSummary, !- Report 4 Name
AllSummary; !- Report 5 Name

!- ===== ALL OBJECTS IN CLASS: OUTPUT:TABLE:MONTHLY =====

Output:Table:Monthly,

FanSplit, !- Name
3, !- Digits After Decimal
Air System Cooling Coil Total Cooling Energy, !- Variable or Meter 1 Name
HoursNonZero, !- Aggregation Type for Variable or Meter 1

Air System Fan Electric Energy, !- Variable or Meter 2 Name
 SumOrAverageDuringHoursShown, !- Aggregation Type for Variable or Meter 2
 Air System Heating Coil Total Heating Energy, !- Variable or Meter 3 Name
 HoursNonZero, !- Aggregation Type for Variable or Meter 3
 Air System Fan Electric Energy, !- Variable or Meter 4 Name
 SumOrAverageDuringHoursShown, !- Aggregation Type for Variable or Meter 4
 Air System Fan Electric Energy, !- Variable or Meter 5 Name
 SumOrAverage; !- Aggregation Type for Variable or Meter 5

Output:Table:Monthly,
 CoilLoads, !- Name
 2, !- Digits After Decimal
 Heating Coil Total Heating Rate, !- Variable or Meter 1 Name
 SumOrAverage, !- Aggregation Type for Variable or Meter 1
 Heating Coil Air Heating Rate, !- Variable or Meter 2 Name
 SumOrAverage, !- Aggregation Type for Variable or Meter 2
 Cooling Coil Total Cooling Rate, !- Variable or Meter 3 Name
 SumOrAverage; !- Aggregation Type for Variable or Meter 3

!- ===== ALL OBJECTS IN CLASS: OUTPUTCONTROL:TABLE:STYLE =====

OutputControl:Table:Style,
 CommaAndHTML, !- Column Separator
 InchPound; !- Unit Conversion

!- ===== ALL OBJECTS IN CLASS: OUTPUT:VARIABLE =====

Output:Variable,
 *, !- Key Value
 Site Outdoor Air Drybulb Temperature, !- Variable Name
 Timestep; !- Reporting Frequency

Output:Variable,
 *, !- Key Value
 Cooling Coil Total Cooling Rate, !- Variable Name
 Hourly; !- Reporting Frequency

Output:Variable,
 *, !- Key Value
 Heating Coil Air Heating Rate, !- Variable Name
 Hourly; !- Reporting Frequency

Output:Variable,
 living_unit1, !- Key Value
 Zone Air Temperature, !- Variable Name
 Timestep; !- Reporting Frequency

Output:Variable,
 living_unit1, !- Key Value
 Zone Thermostat Heating Setpoint Temperature, !- Variable Name
 Timestep; !- Reporting Frequency

Output:Variable,
 living_unit1, !- Key Value
 Zone Thermostat Cooling Setpoint Temperature, !- Variable Name
 Timestep; !- Reporting Frequency

Output:Variable,
 *, !- Key Value
 Site Outdoor Air Relative Humidity, !- Variable Name
 Timestep; !- Reporting Frequency

Output:Variable,
 *, !- Key Value
 Zone Air Relative Humidity, !- Variable Name
 Timestep; !- Reporting Frequency

Appendix O. FMU File (C code)

```
// Basic built in files
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
// Project linker files
#include "FMU_Header_Files\Joe_ep_fmu.h" // Need to be in same folder as fmi header files to link properly
#include "Parser_Files\xml_parser.h" // Put in folder for organization

ModelDescription* md; //creates md from xml file. Requires Parse Files

#include<winsock2.h>
WSADATA wsa;
SOCKET s;
int isWarmupFlag = 1;
int nextStringIndex = 0;
char nextString[2048];

char* getNextString(char myMsg[2048])
{
    //printf("\n===== In getNextString Function =====\n");
    int currentIndex = nextStringIndex;
    memset(nextString, '\0', 2048); // resets memory to null
    while (!(myMsg[currentIndex] == '\r' && myMsg[currentIndex + 1] == '\n'))
    {
        //printf("currentInd: %d\n", currentIndex);
        nextString[currentIndex - nextStringIndex] = myMsg[currentIndex];
        currentIndex = currentIndex + 1;
    }
    //printf("diffInd %d\n", currentIndex - nextStringIndex);
    if ((currentIndex - nextStringIndex) == 0)
    {
        nextStringIndex = 0;
        return '\0';
    }
    nextString[currentIndex - nextStringIndex] = '\0';
    //printf("nextString = %s\n", nextString1);
    //printf("<===== End of getNextString Function\n");
    nextStringIndex = currentIndex + 2;

    return nextString;
    //return currentIndex;
}
```

```

int myLineReader(SOCKET s, char server_reply[2048], char buffer[2048])
{
    int currentBufferLocation = 0;
    int recievedReplyLength;
    while ((recievedReplyLength = recv(s, server_reply, 2048, 0)) > 0)
    {
        server_reply[recievedReplyLength] = '\0';
        if (currentBufferLocation == 0)
        {
            sprintf(buffer, "%s", server_reply); // or else something happens
        }
        else
        {
            sprintf(buffer, "%s%s", buffer, server_reply);
        }
        currentBufferLocation = currentBufferLocation + recievedReplyLength;
        if (currentBufferLocation >= 2048)
        {
            fprintf(myOutputLog, "bufferSize too large. Truncated incoming data.\n");
            break;
        }
        if (server_reply[recievedReplyLength - 2] == '\r' && server_reply[recievedReplyLength - 1]
        == '\n') // && bufferSize >= 2
        {
            //fprintf(myOutputLog, "Recieved end packet. \n");
            break;
        }
    }
    return 1;
}

/*-----*/
// FMI FUNCTIONS: Platform, Version, Logging ...
/*-----*/
const char* fmiGetTypesPlatform()
{
    return fmiPlatform;
}

const char* fmiGetVersion()
{
    return fmiVersion;
}

fmiStatus fmiSetDebugLogging(fmiComponent c, fmiBoolean loggingOn)
{
    return fmiOK;
}

```

```

/*-----*/
// FMI DATA EXCHANGE FUNCTIONS: fmiGet__()
/*-----*/
fmiStatus fmiGetReal(fmiComponent c, const fmiValueReference vr[], size_t nvr, fmiReal value[])
{
    fprintf(myOutputLog, "\n===== fmiGetReal
=====\\n");
    fprintf(myOutputLog, "MASTER (E+) GETTING VARS FROM SLAVE (fmu)\\n");

    unsigned int i;
    for (i = 0; i < nvr; ++i)
    {
        ScalarVariable* myInst = getVariable(md, vr[i], elm_Real); // pulls specific var from md
        const char* thisVarName = getName(myInst); // gets name of var from md

        fprintf(myOutputLog, "SENT var to E+\\n");
        fprintf(myOutputLog, "  Name: ----- %s \\n", thisVarName);
        value[i] = my_values[vr[i]]; // replaces E+ value from FMI's stored values
        fprintf(myOutputLog, "  Value: ----- %.2f \\n", (float)value[i]);
        fprintf(myOutputLog, "  Value Reference --- '%d' \\n", vr[i]);
    }
    fflush(myOutputLog);
    return fmiOK;
}

/*-----ONLY ABOVE USED ^^-----*/
fmiStatus fmiGetInteger(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, fmiInteger value[])
{
    return fmiError;
}

fmiStatus fmiGetBoolean(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, fmiBoolean value[])
{
    return fmiError;
}

fmiStatus fmiGetString(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, fmiString value[])
{
    return fmiError;
}

/*-----*/
// FMI DATA EXCHANGE FUNCTIONS: fmiSet__()
/*-----*/
fmiStatus fmiSetReal(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, const fmiReal value[])

```

```

{
    fprintf(myOutputLog, "\n===== fmiSetReal
=====\\n");
    fprintf(myOutputLog, "MASTER (E+) SENDING VARS TO SLAVE (fmu)\\n");

    unsigned int i;
    for (i = 0; i < nvr; ++i)
    {
        ScalarVariable* myInst = getVariable(md, vr[i], elm_Real); // pulls specific var from md
        const char* thisVarName = getName(myInst); // gets name of var from md

        fprintf(myOutputLog, "RECIEVED var from E+\\n");
        fprintf(myOutputLog, "  Name: ----- %s\\n", thisVarName);
        fprintf(myOutputLog, "  Value: ----- %.2f\\n", value[i]);
        fprintf(myOutputLog, "  Value Reference: -- '%d'\\n", vr[i]);
        my_values[vr[i]] = (float)value[i]; // copys E+ values into the FMI's stored values
    }
    fflush(myOutputLog);
    return fmiOK;
}

/*-----ONLY ABOVE USED ^^-----*/
fmiStatus fmiSetInteger(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, const fmiInteger value[])
{
    return fmiError;
}

fmiStatus fmiSetBoolean(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, const fmiBoolean value[])
{
    return fmiError;
}

fmiStatus fmiSetString(fmiComponent c, const fmiValueReference vr[],
    size_t nvr, const fmiString value[])
{
    return fmiError;
}

/*-----*/
// Creation and destruction of slave instances and setting debug status
/*-----*/
fmiComponent fmiInstantiateSlave(fmiString instanceName, fmiString fmuGUID,
    fmiString fmuLocation, fmiString mimeType, fmiReal timeout, fmiBoolean visible,
    fmiBoolean interactive, fmiCallbackFunctions functions, fmiBoolean loggingOn)
{
    myOutputLog = fopen("../myOutputLog.log", "w"); // creates log file to write

```

```

        fprintf(myOutputLog, "\n===== fmIstantiateSlave
=====\\n");
        fprintf(myOutputLog, "FMU LOCATION: %s\\n", fmuLocation);
        fflush(myOutputLog);

        size_t sizeOfFmuLocation = strlen(fmuLocation); // finds size of location string
        char * fmuFilePath = (char *)malloc((sizeOfFmuLocation + 1) * sizeof(char)); //allocates memory
for file location string
        for (size_t i = 0; i < sizeOfFmuLocation; i++) // switches direction of backslashes to work properly
        {
            if (fmuLocation[i] == '\\') {
                fmuFilePath[i] = '/';
            }
            else {
                fmuFilePath[i] = fmuLocation[i];
            }
        }
        if (fmuFilePath[sizeOfFmuLocation - 1] == '/') // removes final slash if there is one
        {
            fmuFilePath[sizeOfFmuLocation - 1] = '\\0';
        }
        fmuFilePath[sizeOfFmuLocation] = '\\0'; // assures there is an ending null character (OK to have two)

        char * xmlFileName = "modelDescription.xml"; // name of xml in fmu
        //allocates memory for full file location
        char * xmlFilePath = (char *)malloc(sizeof(char)*(strlen(fmuFilePath) + 1 + strlen(xmlFileName) +
1));
        sprintf(xmlFilePath, "%s/%s", fmuFilePath, xmlFileName); // copys fmupath and xml file name for
full xml path
        fprintf(myOutputLog, "XML LOCATION: %s\\n", xmlFilePath);

        md = parse(xmlFilePath); // parses through xml to populate model description, md

        int myNumStates = getNumberOfStates(md);
        int myNumEventIndicators = getNumberOfEventIndicators(md);
        int myNumVars = md->n;
        fprintf(myOutputLog, "myNumStates: ----- %d\\n", myNumStates);
        fprintf(myOutputLog, "myNumEventIndicators: -- %d\\n", myNumEventIndicators);
        fprintf(myOutputLog, "myNumVars: ----- %d\\n", myNumVars);
        fflush(myOutputLog);
        /* CODE TO PRINT CONTENTS OF XML FOR SANITY PURPOSE
        FILE * xmlFile = fopen(xmlFilePath, "r");
        if (xmlFile == NULL)
        {
            fprintf(myOutputLog, "ERROR: No XML file found in FMU\\n");
            // logs error if no xml file in fmu
        }

        char nextCharacter;

```

```

fprintf(myOutputLog, "\n===== XML Content =====\n");
while ((nextCharacter = getc(xmlFile)) != EOF) {
fprintf(myOutputLog, "%c", nextCharacter);
}
fprintf(myOutputLog, "\n");
fclose(xmlFile);
/*END CODE TO PRINT CONTENTS OF XML*/

// removes vars from memory
free(xmlFilePath);
free(fmuFilePath);
fflush(myOutputLog);

//TODO: restructure to find # vars based on modelDescription, md
my_values = malloc(sizeof(float) * 10); // 10 represents # of variables transfered (should change
appropriately)

// TODO: HARD CODED -- WANT SOME WAY AROUND THIS
my_values[4] = (float)25; //starting cooling Start
my_values[3] = (float)23; //starting heating Start
fprintf(myOutputLog, "NOTE:\n");
fprintf(myOutputLog, " - EnergyPlus cannot be one day duration (for warmup distinction).\n");
fprintf(myOutputLog, " - Should look into finding total \"input\" and \"output\" causalities.\n");
fprintf(myOutputLog, " -- Information will change allocation for my_values array.\n");
fprintf(myOutputLog, " - Inaccuracies exist when turning receiving string value to float.\n");
fprintf(myOutputLog, " - Initial conditions (first iteration) hardcoded here. \n");
fprintf(myOutputLog, " -- Will be replaced by socket before E+ reads it.\n");
fflush(myOutputLog);

return my_values; //Seems need to return allocated memory
} // End fmiInstantiateSlave()

fmiStatus fmiInitializeSlave(fmiComponent c, fmiReal tStart, fmiBoolean StopTimeDefined, fmiReal
tStop)
{
fprintf(myOutputLog, "\n===== fmiInitializeSlave
===== \n");

if (tStop - tStart > 86400)
{
isWarmupFlag = 0;
}

if (isWarmupFlag == 0)
{
fprintf(myOutputLog, "----- STARTING SOCKET CONNECTION ----- \n");

```

```

// THIS IS SOCKET STUFF
struct sockaddr_in server;

fprintf(myOutputLog, "Initializing Winsock...\n");
if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
{
    fprintf(myOutputLog, "Failed. Error Code : %d", WSAGetLastError());
    //return 1;
}
fprintf(myOutputLog, "...Initialized.\n");

//Create a socket
if ((s = socket(AF_INET, SOCK_STREAM, 0)) == INVALID_SOCKET)
{
    fprintf(myOutputLog, "Could not create socket : %d", WSAGetLastError());
}
fprintf(myOutputLog, "...Socket created.\n");

//IP information
server.sin_addr.s_addr = inet_addr("192.168.56.101"); //ip address
server.sin_family = AF_INET;
server.sin_port = htons(6789); //port #

//Connect to remote server
if (connect(s, (struct sockaddr *)&server, sizeof(server)) < 0)
{
    fprintf(myOutputLog, "connect error\n");
    //return 1;
}
fprintf(myOutputLog, "...Connected\n");
}
return fmiOK;
} // End fmiInitializeSlave()

fmiStatus fmiTerminateSlave(fmiComponent c)
{
    return fmiOK;
}

fmiStatus fmiResetSlave(fmiComponent c)
{
    return fmiOK;
}

/*----- fmiFreeSlaveInstance() -----*/
// Master FMU is done with the Slave.
/*-----*/
void fmiFreeSlaveInstance(fmiComponent c)
{

```



```

        fprintf(myOutputLog, "\n=====
=====\\n");
        // frees remaining vars from memory
        free(my_values);
        freeElement(md); // frees modelDescription, md

        if (isWarmupFlag == 0)
        {
            printf("--- CLOSING CONNECTION ---\\n");
            fprintf(myOutputLog, "--- CLOSING CONNECTION ---\\n");

            char message[1024];
            sprintf(message, "TERMINATE\\r\\n\\r\\n");
            if (send(s, message, strlen(message), 0) < 0)
            {
                fprintf(myOutputLog, "Send failed\\n");
                //return 1;
            }
            fprintf(myOutputLog, "Data Sent: %s\\n", message);
            closesocket(s); // frees socket
            WSACleanup(); // cleans socket
        }

        fclose(myOutputLog); //closes output file
    }
    /*-----*/
    // DERIVATIVES
    /*-----*/
    fmiStatus fmiSetRealInputDerivatives(fmiComponent c, const fmiValueReference vr[],
        size_t nvr, const fmiInteger order[], const fmiReal value[])
    {
        return fmiError;
    }

    fmiStatus fmiGetRealOutputDerivatives(fmiComponent c, const fmiValueReference vr[],
        size_t nvr, const fmiInteger order[], fmiReal value[])
    {
        return fmiError;
    }

    /*----- fmiCancelStep() -----*/
    // STEPS
    /*-----*/
    fmiStatus fmiCancelStep(fmiComponent c)
    {
        return fmiError;
    }

    fmiStatus fmiDoStep(fmiComponent c, fmiReal currentCommunicationPoint,

```

```

fmiReal communicationStepSize, fmiBoolean newStep)
{
    fprintf(myOutputLog, "\n===== fmiDoStep (%.0f sec)
=====\\n", currentCommunicationPoint);
    unsigned int i;
    for (i = 1; i<5; ++i)
    {
        ScalarVariable* myInst = getVariable(md, i, elm_Real); // pulls specific var from md
        const char* thisVarName = getName(myInst); // gets name of var from md

        fprintf(myOutputLog, "%d: %f ----- %s\\n", i, my_values[i], thisVarName);
        fflush(myOutputLog);
    }
    if (isWarmupFlag == 0)
    {
        // SOCKET CONNECTION VARS
        char mySendMsg[2048], server_reply[2048];
        int recv_size;
        char buffer[2048];

        // ===== Sending Data =====
        fprintf(myOutputLog, "<----- Sending Data ----->\\n");
        printf("<----- Sending Data ----->\\n");

        // Loops through all output vars into string to send
        sprintf(mySendMsg, "UPDATE\\n%.0f\\n", currentCommunicationPoint); //sets start of
outgoing message with timestamp
        for (i = 1; i < 3; ++i) // TODO loop through all "input" causalities
        {
            ScalarVariable* myInst = getVariable(md, i, elm_Real); // pulls specific var from
md
            const char* thisVarName = getName(myInst); // gets name of var from md
            fprintf(myOutputLog, " Preparing to send %s as %f\\n", thisVarName,
my_values[i]);
            fflush(myOutputLog);
            printf("Preparing to send %s as %f\\n", thisVarName, my_values[i]);
            sprintf(mySendMsg, "%s%s\\n%.0f\\n", mySendMsg, thisVarName, my_values[i]);
        }
        sprintf(mySendMsg, "%s\\n", mySendMsg); // adds final \\n to outgoing message
        // Send through socket command
        if (send(s, mySendMsg, strlen(mySendMsg), 0) < 0)
        {
            fprintf(myOutputLog, " Send failed\\n");
            printf("Send failed\\n");
            return fmiError;
        }
        fprintf(myOutputLog, " Data Sent!\\n");
        printf("Data Sent!\\n");
    }
}

```

```

// ----- Recieving Data -----
fprintf(myOutputLog, "-----> Recieving Data <-----\n");
printf("-----> Recieving Data <-----\n");

if (myLineReader(s, server_reply, buffer) < 0)
{
    fprintf(myOutputLog, "    Could not read line properly\n");
    fflush(myOutputLog);
    return fmiError;
}
fprintf(myOutputLog, "    Data Recieved! \n");
fflush(myOutputLog);
printf("Data Recieved\n");

// ----- Parsing recieved data string -----
char* thisHeading;
char* thisTimeString;
char* thisVarName;
char* thisValue;

nextStringIndex = 0;

//finds header -- first expected value
thisHeading = getNextString(buffer);
fprintf(myOutputLog, "    Recieved Header as %s\n", thisHeading);
fflush(myOutputLog);
printf("Recieved Header as %s\n", thisHeading);

if (!strcmp(thisHeading, "SET"))
{
    //finds time value -- second expected value
    thisTimeString = getNextString(buffer);
    int thisRecievedTime = atoi(thisTimeString); //converted string to int
    fprintf(myOutputLog, "    Recieved Time as %d\n", thisRecievedTime);
    fflush(myOutputLog);
    printf("Recieved Time as %d\n", thisRecievedTime);

    while ( (thisVarName = getNextString(buffer)) != '\0') // until nothing is between
    {
        // Expected Variable Name
        // **update for thisVarName happens when testing while loop condition
        fprintf(myOutputLog, "    Recieved %s ", thisVarName);
        fflush(myOutputLog);
        printf("Recieved %s ", thisVarName);

        // gets value reference
        ScalarVariable* myInst = getVariableByName(md, thisHeading);

```

```

        int myValRef = getValueReference(myInst);

        // Expected Value
        thisValue = getNextString(buffer);
        float thisRecievedValue = atof(thisValue); //converted string to float
        fprintf(myOutputLog, "as %f\n", thisRecievedValue);
        fflush(myOutputLog);
        printf("as %f\n", thisRecievedValue);

        // Replacing values
        my_values[myValRef] = thisRecievedValue;
        fprintf(myOutputLog, "  --New  my_values[%d]  =  %f\n",  myValRef,
my_values[myValRef]);
        fflush(myOutputLog);
        //have condition if no matching vr
    }
}
else if (thisHeading == "NOUPDATE\0")
{
    fprintf(myOutputLog, "NOUPDATE\n");
    printf("NOUPDATE\n");;
    fflush(myOutputLog);
}
else
{
    fprintf(myOutputLog, "No expected headers found\n");;
    fflush(myOutputLog);
}
}
return fmiOK;
} // End fmiDoStep()

```

```

/*-----*/
// FMI STATUS FUNCTIONS
/*-----*/
fmiStatus fmiGetStatus(fmiComponent c, const fmiStatusKind s, fmiStatus* value)
{
    return fmiError;
}

fmiStatus fmiGetRealStatus(fmiComponent c, const fmiStatusKind s, fmiReal*  value)
{
    return fmiError;
}

fmiStatus fmiGetIntegerStatus(fmiComponent c, const fmiStatusKind s, fmiInteger* value)
{
    return fmiError;
}

```

```
}

fmiStatus fmiGetBooleanStatus(fmiComponent c, const fmiStatusKind s, fmiBoolean* value)
{
    return fmiError;
}

fmiStatus fmiGetStringStatus(fmiComponent c, const fmiStatusKind s, fmiString* value)
{
    return fmiError;
}
```

Appendix P. XML File for FMI

```
<?xml version="1.0"?>
<fmiModelDescription
  fmiVersion="1.0"
  modelName="Joe_ep_fmu"
  modelIdentifier="Joe_ep_fmu"
  guid="{818642F1-D7D4-4DC7-8549-554862454199}"
  variableNamingConvention="structured"
  numberOfContinuousStates="0"
  numberOfEventIndicators="0">
  <!-- Have to be same as the name in C code-->

  <ModelVariables>
    <ScalarVariable
      name="epSendZoneMeanAirTemp"
      valueReference="1"
      causality="input">
      <!-- name has to match E+ "FMU Variable Name" -->
      <!-- valueReference must be unique -->
      <!-- input/output to/from the slave -->
    <Real />
    </ScalarVariable>
    <ScalarVariable
      name="epSendOutdoorAirTemp"
      valueReference="2"
      causality="input">
    <Real />
    </ScalarVariable>
    <ScalarVariable
      name="epGetStartHeating"
      valueReference="3"
      causality="output">
    <Real />
    </ScalarVariable>
    <ScalarVariable
      name="epGetStartCooling"
      valueReference="4"
      causality="output">
    <Real />
    </ScalarVariable>
  </ModelVariables>

  <Implementation>
    <CoSimulation_StandAlone>
      <Capabilities
        canBeInstantiatedOnlyOncePerProcess="true"
        canNotUseMemoryManagementFunctions="true" />

```

```
</CoSimulation_StandAlone>  
</Implementation>  
</fmiModelDescription>
```