Greenheater

Bringing intelligence to an electric heater

Nicolas Badoux
Grand-Rue 43
1347 Le Sentier, Switzerland
n.badoux@hotmail.com

ABSTRACT

Since a few years, smart thermostat are getting readily available and inexpensive. Providing energy savings, data visualizations and remote actuations capabilities. They offer new possibilities as well as monetary savings. They are replacing the old thermostat and plug themselves on the cables providing actuation of the heating system.

In this paper, I present Greenheater, as smart thermostat for a stand-alone electric heater with the goal of providing energy savings and remote actuation. Greenheater provides automatic temperature management, keeping the room in the selected temperature range when you are present as well as energy saving by letting the temperature drop during the time where you are absent. To avoid discomfort, it automatically preheats the room so that the temperature is in the desired range when you get back.

To achieve this goal, it uses a thermal model of the room as well as occupancy analysis.

Keywords

greencomputing; smart thermostat; energy saving; thermal modelling

1. INTRODUCTION

Due to the economic incentive as well as the goal of reducing our carbon footprint on the environment, we need to decrease our energy consumption. Space heating still accounting for around 40% of the residential sector energy use in the U.S. [13], it is an important field to optimize. Different techniques to make a house more efficient already exists and are deployed, ranging from improving the thermal isolation to increasing heating facilities efficiency as well as changing the heating policy and behavior.

This last method has been particularly studied these past years through smart thermostat research. Thanks to the smart homes and the Internet of Things (IoT) development, we are able to acquire lots of data about diverse parameters like occupancy and temperature. Previous work has taken a look at these data, providing analysis that is further used to fine tune the heating system in a goal of energy savings.

Programmable thermostats have been around for multiple decades, They let the user define different preset of different time, for example lowering the temperature between 9AM and 15PM during weekdays. However, they suffer from multiple drawbacks. Programmed with a strict schedule, they don't adapt to the changing schedules of the house inhabitants. In response, the users use conservative settings for the thermostat or not programming it at all to avoid discomfort but avoiding at the same time any energy savings. [10]

Portable heater, as used in this paper are a common household item. In 2000, only 26% of U.S. household reported owning home but this number grow to 60% in 2010. These heaters are not the main source of heating for the household in most cases. They are used as auxiliary heating when a short and precise heat burst is needed. [11]

For this project, I have implemented a smart thermostat for my particular room, behavior and situation. Not having central heating, as the rest of the house, arriving directly into my room, I have been using a stand-alone electric heater. In comparison with prior research papers (2.1), which are mostly concerned with controlling a whole house form a single thermostat, I aim to control only one device having direct effect on a small space. This offers new opportunities as well as new challenges.

Greenheater is a self-programming thermostat designed to control an electric heater. It monitors occupancy through the user's Google Calendar as well as scanning the home Wi-Fi network where personal devices are connected. The ground idea for energy savings come from the fact that heat dissipation is proportional to the temperature difference between two mediums. Greenheater therefore leverages the periods of absence to let the temperature fall, reducing this way, heat transfers through the walls. To avoid user discomfort, Greenheater needs to adjust the temperature back to the desired range before the user comes back home. To do that, my system uses past temperature data to construct an approximate thermal model of the room. This model let an algorithm predict the time Greenheater needs to restart the heater to get the room to a particular temperature at a particular time. For prediction and building the thermal model, the system uses temperature data from 3 sensors, one on the outside of my room, one in the apartment next to my room and the last one reporting the actual temperature in my room. To improve predictions, it also acquires weather prediction from an online weather forecast service

08-540 Greencomputing Spring 2016, Carnegie Mellon University, Pittsburgh PA, USA

DOI: 10.1145/1235

(wundergound.com).

In this paper I also present an evaluation of the energy savings, insights on the duration of training needed for the thermal model as well as solutions to problems that I encountered that could help some future research.

2. BACKGROUND AND RELATED WORK

2.1 Thermostat

Programmable thermostat exists since over a century, Honeywell producing it first setback thermostat in 1906. [1] Smart thermostat are much more recent, Nest being the first company getting greater prominence in the public in 2010. Research on the subject dates back almost twenty years. M. C. Mozer ran a simulation, in 1997, to show the interest of adaptive control of heating systems based on occupancy. His Neurothermostat was the first to bring machine learning to thermostat. [9]

M. Gupta et al. studied occupancy detection and prediction based on GPS data provided by the user's phone. They take advantage of long commuting time to adjust the room temperature before the user's arrival. Their simulation provided up to 7% of energy savings over an HVAC system. [7]

G. Gao et al. used sensors similar to those in security system to monitor occupancy more closely and took a novel approach, letting the users adjust with a knob the balance between energy savings or comfort. [3]

J. Lu et al. developed a smart thermostat that used occupancy sensors to train a Hidden Markov Model to predict occupancy and be able to launch the heater at the right time. They optimize their thermostat to use two different heating mode to be able to heat quickly or slowly depending on the certainty of the occupancy prediction. [6]

J. Scott et al. developed and deployed in 5 homes PreHeat, a smart thermostat that acquire occupancy data from RFID tags and predict the future occupancy probability based on past data. The main contribution of this paper is a real deployment and evaluation of an occupancy-based thermostat. [12]

2.2 Occupancy detection

As mentioned in these previous papers, a lot of work is based on occupancy detection. This field of study have seen lots of improvements and new ideas these past years with the development of new sensors and decrease in price of existing ones. My particular behavior as a college student allowed me to do some simplification on my model, since I am either on campus or home and in my room. Because of that, I did not consider further occupancy detection and found sufficient a basic scheme.

2.3 Thermal modelling

Thermal modelling is a field older than the smart thermostat and quite well understood. Heat transfer equations are well described in literature but their granularity is too small for our use. They need particular heat conductivity coefficients for materials, precise architecture as well as precise air movement. [5] K. Mohamad created a Matlab tool to construct a thermal model but that still require a lot of parameters that the user probably does not know. [8]

Avoiding physical measurements of the thermal envelope, Greenheater try to extract a thermal model from past data. Greenheater thermal modelling technique is quite similar to the one from C. Ellis et al.[2]. They developed and implemented a room-to-room thermal model to accurately predict the temperature in different parts of a building from temperature traces. They used curve fitting on past sensors data to find coefficient of a theoretical model that they conceived, and achieved less than 1°C error when predicting up to 6 hours in the future. In their paper, they also simulate energy savings when using their model instead of a linear heating rate. [2]

3. DEPLOYMENT

Greenheater is deployed in my personal room. I acquired data from end of February to the April 2016 and evaluate it during the month of April 2016.

3.1 Room and Building Characteristics

My room is located in Pittsburgh, PA, USA in the middle floor of a three story house built in the 19th century. As mentioned before, the house is heated through a central heating system running on natural gas but my room does not have an exit vent. The room is approximatively 18m^2 with a ceiling height close to 3 m . $\frac{2}{3}$ of the wall area in contiguous to the rest of the house, leaving $\frac{1}{3}$ that contains two windows. The walls are $\sim$ 12 cm thick when facing other part of the house and $\sim$ 25 cm when facing outside.

3.2 Electric heater

My electric heater is a Lasko 753500 that came with my room. It has two power modes at 1500W and 900W, however due to technical difficulties, my system only use the second one. It also has an automatic mode, which was used to determine the baseline power consumption. In automatic mode, the user decides the desired temperature and the heating is managed in order to stay in a range of 1°C around the temperature preset. When just keeping track of the temperature but not providing heating, the automatic mode consumes around 2W.

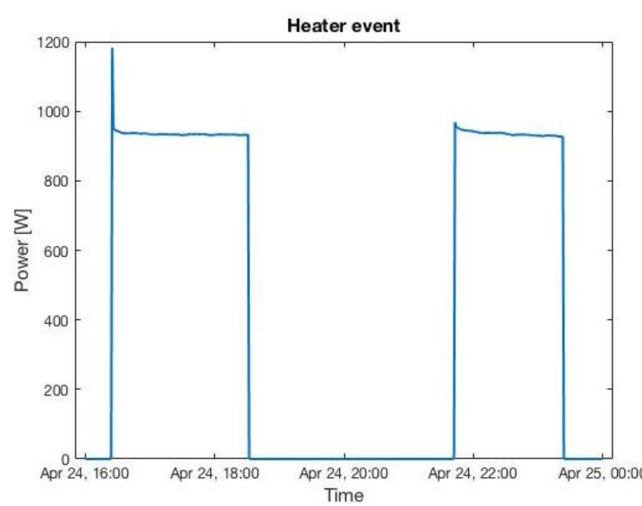

Figure 1: Power draw during a heater event. We can see a spike at the begining but after that the energy consumption stay constant.

The heater is plugged into a Belkin WeMo Insight Switch. The WeMo switch is a publicly available Wi-Fi connected electric switch. Of the shelf, it offers management through its own app and the IFTTT protocol. The app and the switch communicate over HTTP. The protocol was reverse engineered by Joshua Gluck and thanks to his work I was able to turn the WeMo switch off and on and acquire the power consumption. The plug in itself consumes around 1.3W and costs around 50$.

3.3 Temperature sensors and main hub

My thermal model use three temperature data traces, the outside temperature, the temperature in the close apartment environment and the temperature inside the room. To acquire these data I use three Oregon Scientific THN132N wireless sensors, priced at $\sim$ 18$ each. They are meant to communicate with a base station over the RF433 MHz frequency and are powered by AA batteries. I had to change the battery of the external sensors after one winter month. Looking back, I am not sure that it was the battery fault but maybe only obstruction of the receiver antenna.

Instead of a base station, I have a Raspberry Pi Model B used as a main computing unit. Attached to it, is a Wi-Fi USB dongle allows it to communicate with the WeMo switch as well as a RF433 MHz receiver for the communication with the temperature sensors. The RF433 MHz receiver is connected to the GPIO pin of the Raspberry Pi and communicate with the temperature sensors up to a range of $\sim$ 10m. The Raspberry Pi consume around 1.5W of electricity.

Thanks to the work of disk91 and 1000io reading and decoding value of the Oregon sensors was as simple as "plug and play". The overall CPU load of the system stay under 5% leaving the opportunity to use the Raspberry Pi for other tasks.

4. BUILDING A THERMAL MODEL

4.1 Theory

Heat transfer is composed of 3 mechanisms, radiation (transfer through electromagnetic waves, ex: sunlight), convection (transfer through mass motion of a fluid, ex: heated air rising) and conduction (transfer through physical contact). Only radiation and conduction play a role in the case of my room. Radiation is negligible because my room receive little sunlight and it is lesser than other unaccounted variable (leaving the door open for example).

The heat transfer through conduction is modelled by :

$$Q = \frac{kA(T_{hot} - T_{cold})}{d}$$

where Q represent the heat transfer per unit of time [W], k the thermal conductivity of the material $[\frac{W}{mK}]$, A the heat transfer area [m^2] and $T_{hot} - T_{cold}$ the temperature difference between the two materials [K]. We can regroup the constants in a global constant x and have this equation:

$$Q = x\Delta T$$

Looking at all the energy losses in the room we get:

$$Q = x_1(T_{room} - T_{outside}) + x_2(T_{room} - T_{apartment})$$

where x_1 is a constant specific for the area between the outside and the room and x_2 between the room and the rest of the house. By the first law of thermodynamics (abstracting all other energy interaction), the total energy change of the room is modelled by :

$$Q = x_1(T_{room} - T_{outside}) + x_2(T_{room} - T_{apartment}) - G_{heater}$$

where G_{heater} is the energy gain brought by the heater per unit of time. The heat capacity equation give us:

$$Q = C\Delta T$$

where C is depend on the mass and the heat capacity of the material and therefore constant for the room. Now if we look at the temperature inside room at at time $t + d$:

$$T_{room}(t + d) = T_{room}(t) + \frac{Q}{C}$$

or equivalently:

$$\begin{aligned} T_{room}(t + d) = T_{room}(t) &+ \frac{x_1}{C}\int_t^{t+T} \Delta T_{outside}(\tau)d\tau \\ &+ \frac{x_2}{C}\int_t^{t+T} \Delta T_{apart}(\tau)d\tau \\ &- \int_t^{t+T} \frac{G_{heater}(\tau)}{C}d\tau \end{aligned}$$

where $\Delta T_{outside}$ is the difference of temperature between outside and inside the room, equivalently for ΔT_{apart}. Rewriting the constants:

$$\begin{aligned} T_{room}(t + d) = T_{room}(t) &+ c_1\int_t^{t+T} \Delta T_{outside}(\tau)d\tau \\ &+ c_2\int_t^{t+T} \Delta T_{apart}(\tau)d\tau \\ &+ c_3\int_t^{t+T} G_{heater}(\tau)d\tau \end{aligned} \tag{1}$$

4.2 Implementation

For this paper I have tested different methods to find the coefficients c_1, c_2 and c_3. I have implemented a support vector machine for regression in Python with the help of sklearn and pandas but without success due to time constraints and technical difficulties.

My system at the time of this report uses the `lsqcurvefit` of the Matlab Machine Learning toolbox. `lsqcurvefit` try to fit the coefficient of a function using gradient descent to minimize the squared of the error between the training and the predicted data. This approach has the advantage of the simplicity but also presents some drawbacks. For example, it is not an iterative prediction, meaning that if we want to find the coefficients at time $t + d$ we can only use the data available before and at time t. Because of that I had to make assumption to rewrite equation 1. I assumed that the temperature of the rest of the house to stays constant over the time interval d. Obviously, this is not totally true but from the data trace (see Figure 6) I have collected it is a reasonable assumption. Since when predicting for time $t + d$ we do not have access to the room temperature in the d interval, I assumed it to stay constant over this interval. Again, this is clearly false but I have achieved reasonable prediction with this model and was not able to make a better model (SVM or neural network) function properly.

The Equation 1 becomes:

$$\begin{aligned}T_{room}(t+d) = T_{room}(t) + c_1 \int_t^{t+T} T_{outside}(\tau) - T_{room}(t) d\tau \\ + c_2(T_{room}(t) - T_{apartment}(t)) \\ + c_3 \int_t^{t+T} G_{heater}(\tau) d\tau\end{aligned}$$

A further simplification is to assume the energy gain of the heater to be constant in time (see Figure 1):

$$\begin{aligned}T_{room}(t+d) = T_{room}(t) + c_1 \int_t^{t+T} T_{outside}(\tau) \\ - T_{room}(t) d\tau + c_2(T_{room}(t) - T_{apartment}(t)) \\ + c_3 \int_t^{t+T} \mathbb{1}_{heater\ running\ at\ time\ t} G_{heater} d\tau\end{aligned} \tag{2}$$

where $\mathbb{1}_{heater\ running\ at\ time\ t}$ is the indicator function, valued at 1 if the statement is true and 0 otherwise.

This equation only contains value known at time t apart from the future outdoor temperature and how long the heater will be running during the interval. The weather forecast that Greenheater fetches from wunderground.com gives us the first data. The percentage of time the heater will be running will be iteratively increased by our prediction algorithm until finding a correct value.

To predict the right time to start the heater, my program takes the desired room temperatures at a time t and iterates over growing values of d to find the time interval d for which we would have to run the heater continuously. We have found this interval as soon as running the heater for the whole d interval would match the desired temperature at time $t + d$.

4.3 Evaluation

My system samples data from the WeMo switch and the temperature sensors every minute and computes coefficients for d ranging from 5 minutes to 5 hours with a 5 minutes interval. Training is done once on a 7 days period. The coefficients are then evaluated on an independent 10 days period. Greenheater achieves an 90th percentile of error under 0.5°C on a prediction 2 hours in the future. This result matches previous work done by C. Ellis with Matchstick [2].

4.3.1 *Length of training*

I also evaluated the length of training. As we can see in Figure 3 and could expect, the error decreases as the training time increases. I was not able to lengthen the training time enough to find a plateau where additional training time would not improve prediction.

5. THERMOSTAT POLICY AND OCCUPANCY DETECTION

The goal of the policy is to save as much energy as possible without creating any discomfort. Some previous works have studied how the user feels temperature changes and how far we can go without creating discomfort.[4] For the purpose of this work, we will take a basic policy consisting of letting the user fixing a comfort temperature and stick to it without trying to find the users comfort temperature. However, during evaluation we will keep in mind that if we

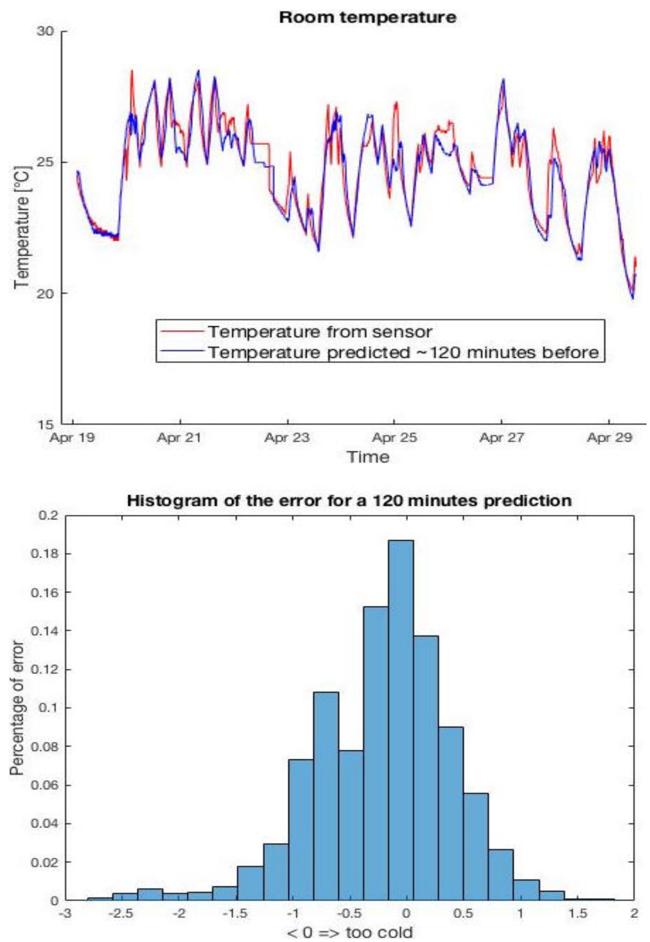

Figure 2: Evaluation of an 8 days training, top: curve predicted in blue, real data in red. bottom: histogram of the norm of the error

miss the user fixed temperature by a tenth of a degree, he will probably not notice it..

The basic idea is to let the temperature decrease during time of absence and increase it again before the user comes home. In addition to this, we will allow temperature to lower during the user sleep. My implementation also takes advantage of my personal behavior. As a college student, I wake up as late as possible before class, therefore spending almost no time in my room between wake up time and the time I leave to go on campus. Taking advantage of that, Greenheater let the temperature decrease as soon as the user go to bed and set it back to the chosen value only for the time he get back from work. A simplified overview is presented in Algorithm 1. That is a simplification that could be considered as problematic for different behaviors (some users want to stay in their room at a comfortable temperature before leaving the room).

5.1 Implementation

Again this project was thought by myself for my particular situation and takes advantage of different particularities of my schedule and behavior that will be detailed shortly.

During the planning of this project, I realized that my behavior was closely matched by the events in my Google

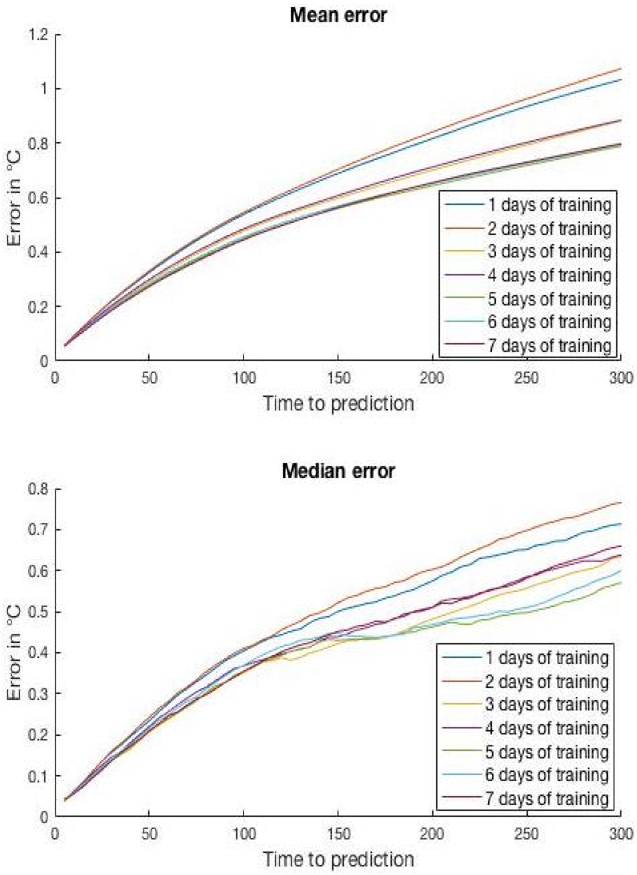

Figure 3: Length of training versus the average and median error and the prediction length [minute]

Calendar. We could resume my day this way, leaving home as late as possible before class (first event of the day) and getting back just after the last class or social event I have on campus. The end time of the last event in my calendar represents accuratly enough the time I will get home (commuting time negligible). We miss certain periods like grocery shopping or chatting with a friend after an event on campus. These events however are quite rare and does not last long. I also never go home between classes if I don't have at least 3 hours of free time. The end time of the last event or the end time of an event before a long break is fetched every 10 minutes via the Google Calendar API and processed on the Raspberry Pi. Living close to campus (biking in ~ 8 minutes), using GPS data as in M. Gupta et al. [7] would not let my heater enough time to heat the room.

To detect sleeping time, I also take advantage of my particular behavior. I use a NFC tag to set up my alarm every night just before going to bed. One important disadvantage of this method is that the thermostat will not detect that it can go to a lower temperature setting if I plan to sleep in and don't set an alarm. We could remedy to that by having a pressure or occupancy sensor near the bed but I did not had enough time to try and implement this method. To obtain the time of the next alarm, I have developed a basic Android app for the sole purpose of doing that. It subscribes to a broadcast receiver from Android OS, notifying me when the time of the next alarm changes. The app communicates

```
while True do
    if not sleeping then
        if present then
            keep the room at the desired temperature;
        else
            if time of return known then
                heat to achieve desired temperature at
                time of return;
            else
                keep the room at the desired
                temperature;
            end
        end
    else
        if time of return known then
            heat to achieve desired temperature at time
            of return;
        else
            keep the room at the desired temperature;
        end
    end
end
```

Algorithm 1: Heating procedure; note: time of return can also represent the wake up time

over HTTP POST request with a PHP script running on top of Apache2 on the Raspberry Pi.

During the development of this app, I encountered multiple networking issues. For example, my phone was not able to access the external IP address from the inside of the home network. To counter that, the app automatically checks if it connected to the home Wi-Fi network and if it is the case use the home subnet IP address of the Raspberry Pi.

Greenheater implement a second occupancy detection technique to be sure to detect if the user is home. It sends multiple ping requests to the user's phone. If the phone is connected to the home network, the user is home and the ping requests will return successfully. If the ping requests don't return successfully, Greenheater will assume that the user is absent. This detection method properly backfired because my home router offers two Wi-Fi networks, one over the 5Ghz and the other on the 2.4Ghz band. The phone being configured to use both and the Raspberry Pi being constantly on the same, some misdetections happened. Deactivating one of the two Wi-Fi networks on the phone solved this problem, however at the cost of some Wi-Fi capabilities.

5.2 Evaluation

This system performed really well and I did not notice any time where I was home and Greenheater not expected me to be. Over a week, I was away when the thermostat expected me home for a maximum aggregate of 2 hours, which represent around 1% of miss time. Even if this approach worked well in my case, it is highly probable that it would not work for others. As a college student is well planned and I enter most of my events in my Google Calendar. This is possible because my week a quite repetitive, going to church on Sunday morning, classes and predefined friends meetings. Because of the good results obtained by this basic method, I did not investigated any other predictive or machine learning occupancy prediction method.

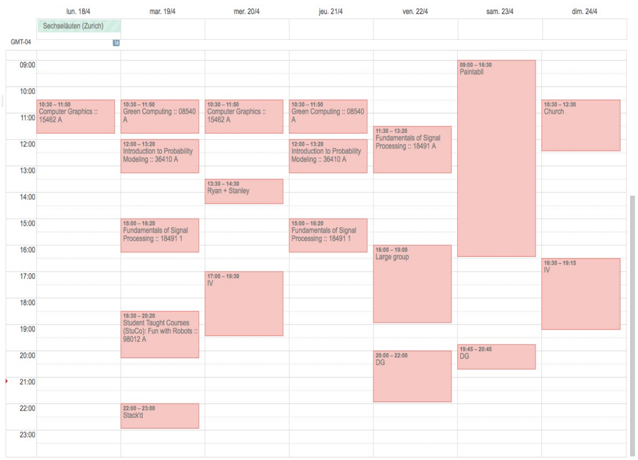

Figure 4: Typical week, weekend and evening activities change from week to week but a strong pattern is still present.

6. ENERGY SAVINGS

6.1 Baseline

I defined the baseline as the heating preset from my heater as it was what I was using before beginning this project. You can choose a temperature and it automatically keeps the room in a range of 1°C around this temperature. Running this preset for 5 days gave a baseline of 10.83KWh per day.

Obviously, the consumption depends a lot on the environment, therefore comparisons are complicated. During the 5 days, the outdoor temperature varied between 20°C and 5°C. The apartment temperature was almost constant around 19°C. and the preset temperature was 24°C. As we can see in Figure 5, the room temperature oscillate between 23°C and 25°C. The values in the middle are strange but I was not able to find any discrepancies in the original data. I performed analysis both with and without. The baseline would be of 14.17KWh per day if the flat part was included.

To avoid any errors due to lost of outdoor temperature during this period, the comparison with Greenheater will be done without this data period.

Outdoor temperature	Automatic	Greenheater
Average	15.35	15.00
Median	15.50	13.50
Standard deivation	4.17	4.89

Table 1: Comparison metrics for the outdoor temperature during the periods where the baseline and Greenheater were running.

6.2 Greenheater

Greenheater was run a few days after the automatic mode. Since the outdoor temperatures play an important role in the energy saving I tried to have a data period having similar temperatures as the one when the automatic ran. Some comparison metrics for the two periods are displayed in Table 1. The two periods are identical but stay in a similar temperature range. To avoid any bias in favor in Greenheater, I also set the desired temperature 1°C higher than during automatic mode.

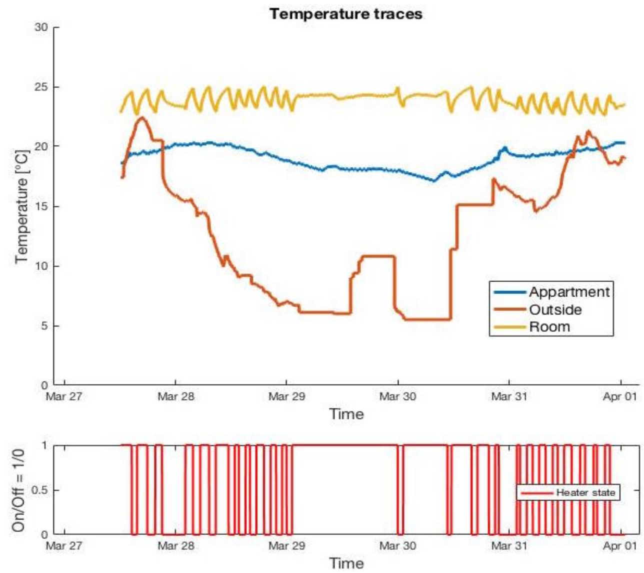

Figure 5: Temperature traces [°C] and heater state during automatic mode. The outdoor temerature shows the absence of some data points.

As expected, Figure 6 shows clearly the room temperature oscillating on a wider range than before. We can also distinct the period of absence or sleep where the temperature decrease under the set temperature during a few hours.

For the energy consumption computations, I added to the power consumption reported by the WeMo switch the constant power draw of the Raspberry Pi and the WeMo plug. After ignoring the days where the outside temperature differs too much, I get a daily energy consumption of 8.79KWh. In comparison with the baseline of 10.83KWh, it is 19% energy savings. This number is consistent with previous work done in this field. [6]

I will not try to compute yearly savings estimation. Extrapolating yearly saving from a system used during 6 consecutive days in a unique season of the year seems to be too hazardous. As a rule of thumb, I do not expect any savings during the summer as the heater is not running most of the time. I would expect similar energy savings during winter and autumn. However, if the winter is really cold, Greenheater might not be able to provide much energy savings as it would need to run continuously anyway.

19% energy comsumption reduction could seem small, but we must keep in mind that it is energy savings applicable to a sector with a high energy comsumption.[13]. When looking only at the electricity comsumption, space heating represents 12%. [14]

The resulsts would certainly be smaller if the 200 years old house was better isolated. In comparison with other methods, like improving thermal isolation, Greenheater leads to lower savings but is much easier to install and way cheaper. As always, when multiple systems are used together, the overall savings is reduced between each other.

The thermal model prediction was successful during the time when Greenheater was running. I did not notice any big difference between the set temperature and the temperature when I reached my room after work.

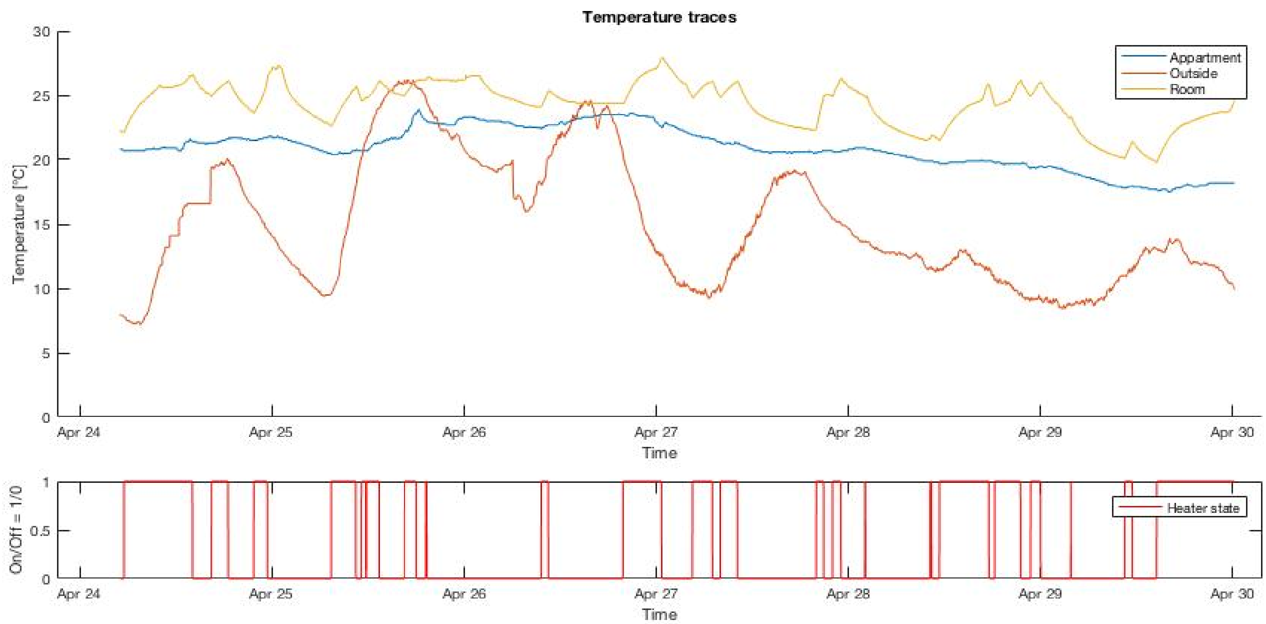

Figure 6: Temperatures traces and heater state during Greenheater mode. Yellow = room, blue = apartment, orange = outside, red = heater.

7. USER INTERFACE

A thermostat needs to be available to the user and the user needs some way to interact with it. Even if Greenheater run on its own without problem, it is very likely that the user will at some point want to change the desired temperature or the range in which it can fluctuate. To fulfil these requirements, my system disposes of a basic website offering buttons to change the comfort temperature, the range as well as one to stop Greenheater and use the heater manually. The webpage is password protected to avoid change by an unauthorized person that would create much discomfort.

No user study was conducted to compare different interfaces in this paper. The actual interface is close to a basic thermostat and therefore is expected to fulfil its duty but without fanciness or elaborated design. The Android app used to retrieve the time of the next alarm is not able yet to control Greenheater due to time constraints. However, I plan to offer this option in the future. This would also make possible an app for Android wear, letting you control the thermostat from a smartwatch for example.

Greenheater not being meant to be a commercial product, it is not compatible with home automation technologies like IFTTT or Apple Homekit. This would however be considered in the case of further developments of the interface and interaction possibilities.

Most configurations parameters (minimum duration for a break for you to go home for example) are for now only accessible through command line input. As we cannot expect from a normal user to know how to manipulate a terminal, these parameters would probably be added to a configuration pane of the webpage in the future.

8. DIFFICULTIES ENCOUNTERED

Almost every home having a Wi-Fi network, connectivity inside a home should not be a problem. However, I encountered multiple problems, mostly due to the fact that our routers are not conceived with remote access in mind. I spent lot of time trying to access the external IP address within the home network. During the deployment of Greenheater, my apartment also had to change router due to failure. As it did not provide an easy interface to transfer parameters, reconfiguration of the router and system took multiple hours, still discovering bugs even a few days later.

I would recommend a fellow researcher to directly implement a way to receive notifications if the system fails. At best provide directly the system with failsafe (automatically start scripts on boot for example). During my work, I encountered two power failures that took me a few hours to notice each time. It took me also more than a day to notice than the outdoor sensors fell of his initial placement and was out of reach of the receiver.

Even if lots of progress have been made in the IoT, setting up the electric switches I used is still cumbersome and would probably stop most users of installing one. There is still a lot of options for improvement to make it more ”plug and play”.

I would also recommend developing the system on a modular approach. As I encountered multiple bugs in certain parts of the system (Apache server not responding, receiver obstructed and not able to receive data from the sensors, file corruption due to a power outage that stopped a writing process...), having a modular system let part of the system running even if some other part is failing.

As the semester went forward and the weather temperature rose, I actually did not really needed the heater, since I can live happily in a room at 18°C. To be able to acquire data, I had to set the desired temperature way higher than my comfort, actually using way more energy that I would have normally used for the purpose of my research.

9. LIMITATIONS AND FUTURE WORK

As mentioned multiple times in this paper, I took often advantages of particularities of my situation, behavior and sensitivity. For example, waking up in a cold room does not disturb me but can be a great discomfort for other persons.

Greenheater was developed with the idea of a unique room. Scaling it to a house would require a lot of changes if we would like to control everything from a unique thermostat. However, if every room dispose from its own heating system, Greenheater could be easily deployed by multiplying the number of sensors and actuators and considering each room as individual. This approach would nevertheless miss a lots of fine tuning possibilities offered by the management of a whole house with actuators acting cooperatively.

Greenheater was also constructed with a particular type of user in mind: people with strict and repetitive schedules that are available in form of a calendar. Extending to people without these characteristics would ask a new and much more complex occupancy prediction algorithm.

My system would probably also greatly profit of a thermal model mapping the theory more closely. By lack of time and experience I was not able to successfully implement a machine learning algorithm that would create an iterative prediction model.

In addition to the automatic mode present on the heater, a manual mode was used before the development of this paper. It is the simple policy "If I am cold, start the heater, if I am to warm, stop it.". This policy would probably yield even better energy savings that Greenheater in comparison to the baseline, however, it imposes discomfort to the user that is hard to evaluate since it is felt differently by each person.

This paper is also centered on an electric heater but it is possible to adapt it to other sources of heating as long as the energy gain offered by the heater are constant in time.

Future work could include developing a policy not for saving energy but to save money by heating at the time where it is the cheapest. In this case, instead of taking advantage of lowering the temperature, the system would rather heat over the desired temperature and let it decrease afterwardds. This approach need data and prediction of electricity prices that were not available to me.

Studying the comfort zone of the end user could also bring new approach and possibilities for energy savings. Like adapting the inside temperature with the outside, a cold spring day probably mean a sweatshirt, offering the possibility to lower the comfort temperature. Or studying if a person can tolerate lower temperature after having done some sports. Having a thermostat knowing the user comfort level can probably yield important energy savings by not having the user increase the set temperature and leaving it this way for a couple of days before replacing it to its normal value.

Greenheater use weather forecast from wunderground.com. I did not had time to evaluate any other weather forecast services or the necessity of one. This could either provide more accurate prediction or show a way of simplifying the system.

I do not live in this room during the summer which explain the absence of A/C unit. However, due to the similar behavior of space heating and air cooling, Greenheater would be able to manage an A/C unit the same way as the heater unit.

As the date of this paper, Greenheater does not provide any way of visualizing the energy saving or temperatures and power traces easily. Offering a visualization panel on the webpage would probably interest some users.

10. CONCLUSION

In this paper I presented and discussed the implementation of Greenheater, a smart thermostat for a stand-alone electric heater that leverages period of absence to achieve energy savings. Greenheater take advantage of the strict schedule of a college student to infer basic occupancy prediction. This policy was successful in my particular case but would be adaptable for other groups of people with similar behavior.

Policy evaluation has shown that the policy used in this project was sufficiently accurate to not cause any discomfort as well as providing energy savings.

We first evaluate the thermal model of the room where the system will be deployed, with the help of past temperatures traces. This thermal model, based on least square curve fitting, achieve a decent accuracy but nothing great. Further experimentations with machine learning algorithms would probably let us decrease the 0.5°C mean error for the 90-th percentile.

By combining multiple services, Google Calendar, Wunderground.com, Android app,... Greenheater is a fully automatic thermostat that adapt to your schedule and environment.

As shown in the evaluation, this project offers 19% of energy saving on a spring day. Energy savings in this range begin to be financially profitable. Thanks to the ease of deployment, similar systems to Greenheater could be deployed quickly in large parts of the residential sectors and provide important advancements toward sustainability goals.

Even if this system was developed with a particular situation and behavior in mind, it would be extensible to other situations, as long as certain assumptions are maintained.

11. MORE INFORMATIONS

If you want more information, feel free to contact the author or take a look at the Github repository of this project.

12. ACKNOWLEDGMENTS

I want to acknowledge once again Joshua Gluck, disk91 and 1000io whose codes have been really helpful to develop this project. I would also like to thanks Professor Yuvraj Agarwal for his guidance and availability throughout this project as well as Julien Lamour for proofreading this paper.

13. REFERENCES

[1] 100 years of programmable thermostats. http://www.prothermostats.com/history.php.

[2] C. Ellis, M. Hazas, and J. Scott. Matchstick: A room-to-room thermal model for predicting indoor temperature from wireless sensor data. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 31–42. ACM, 2013.

[3] G. Gao and K. Whitehouse. The self-programming thermostat: optimizing setback schedules based on home occupancy patterns. In *Proceedings of the First*

ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, pages 67–72. ACM, 2009.

[4] S. Karjalainen. Thermal comfort and use of thermostats in finnish homes and offices. *Building and Environment*, 44(6):1237–1245, 2009.

[5] T. Kusuda. Fundamentals of building heat transfer. *JOURNAL OF RESEARCH of the National Bureau of Standards*, 82(2), 1977.

[6] J. Lu, T. Sookoor, V. Srinivasan, G. Gao, B. Holben, J. Stankovic, E. Field, and K. Whitehouse. The smart thermostat: using occupancy sensors to save energy in homes. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 211–224. ACM, 2010.

[7] K. L. Manu Gupta, Stephen S. Intille. Adding gps-control to traditional thermostats: An exploration of potential energy savings and design challenges. In *Pervasive Computing*, pages 95–114. Springer Berlin Heidelberg, 2009. 7th International Conference.

[8] M. K. Mohamad. Developing a thermal model for a residential room using simulink/matlab, June 2012.

[9] M. C. Mozer, L. Vidmar, R. H. Dodier, et al. The neurothermostat: Predictive optimal control of residential heating systems. *Advances in Neural Information Processing Systems*, pages 953–959, 1997.

[10] A. Plourde. Programmable thermostats as means of generating energy savings: some pros and cons. *Canadian Building Energy End-Use Data and Analysis Centre, Technical Report CBEEDAC 2003-RP*, 1, 2003.

[11] A. Riedel. US Portable Heater Market Snapshot 2013. http://fr.slideshare.net/AJRat4RMG/2013-us-portable-heater-market-report-46997091.

[12] J. Scott, A. Bernheim Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, and N. Villar. Preheat: controlling home heating using occupancy prediction. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 281–290. ACM, 2011.

[13] U.S. Energy Information Administration (EIA). Heating and cooling no longer majority of u.s. home energy use. https://www.eia.gov/todayinenergy/detail.cfm?id=10271, March 2013.

[14] U.S. Energy Information Administration (EIA). Residential energy consumption survey (recs), *How is electricity used in U.S. homes?* https://www.eia.gov/tools/faqs/faq.cfm?id=96&t=3, April 2015.