Entracker: Energy Tracker for Homes

Sunil Mallya
Department of Computer Science
Brown University
Providence, Rhode Island
sunilmallya@cs.brown.edu

Spring 2011

Abstract

Energy tracking and monitoring in homes has gained momentum in recent years due to its important implications in energy conservation and economic savings. Many existing systems require deployment of large number of sensors to provide energy tracking of individual devices. Many other systems rely on statistical data or user inputs to disaggregate the total energy. In this report we present a cost effective and a single-point pluggable system which can track the energy consumption of individual devices. We also introduce evaluation metrics to verify the power disaggregation. The system uses a single plugin data acquisition unit to detect device events and a power meter to record the total power consumption. We test the system on a real circuit that models a house with different devices. Entracker could account and breakdown over 95% of the total energy. Our estimation of the average device power consumption was close to the rated power wattage of the devices.

1 Introduction and Motivation

In recent years there has been significant interest to build smart energy meters which can monitor the consumption of power for individual electrical appliances in our homes. Researchers have envisioned a device which can detect every electrical appliance in your house and measure its energy consumption.This data can be used to guide the user to understand his energy needs and promote energy conservation. There have been commercial attempts at making such smart meters, but most of these meters are focused on giving user the aggregate consumption of power. This data may not be useful for the user since he is unaware of the of consumption at each device. The power profile containing aggregate consumption may be useful information, but its difficult from the user's perspective to translate this into the actions which could reduce the energy consumption. A lot of these meters today use survey data to infer the individual breakdown of consumption. They use known load profiles of houses and statistical characterization of consumption behaviour to profile the energy. This technique may not be accurate as many of the devices could share the similar load profiles. Its important to understand that not Eve user has the same usage pattern or the same idea of energy conservation. Hence generic systems fail to identify customized needs. An adaptive system which can provide real time analysis of the data with fine granularity would be able to cater to the needs of most users.There also have been systems built which rely on a large number of sensors to track individual energy consumption, but we feel these systems are hard to deploy and may be expensive since every device needs to have its own individual sensor. This motivated us to build a fine grained energy tracker which gives the user disaggregated energy consumption per device.

Most of the electricity in the US is produced in thermal power plants which burn coal. Hence the amount of electricity consumed is directly related to the amount of coal burnt and this contributes to the

CO_2 emissions. The stats from Office of Energy Efficiency and Renewable Energy indicate that an average household in the US consume around $1900 on home utilities and fortunately a large proportion of this can be cut down by identifying both individual and aggregate energy usage patterns. Even a 10% reduction in electricity usage across homes could save millions of dollars. This could also reduce the usage of coal to generate electricity hence consuming the natural resources for a longer period of time. Such systems can also benefit the users by guiding them to detect faulty or misbehaving devices. If a user notices that a certain device's average consumption has significantly risen then this may be an indication for the user to look into that particular device. Also if there is a significant rise in the phantom load that the devices consume it may be an indication of faulty electric lines or insulation faults. Leveraging this disaggregated information, automated feedback systems could be built which providing the user real time feedback or suggestions on emergent behaviours in the system. Such automated systems could also help users to identify time independent energy usages and schedule them to off peak load times, which would help them significantly reduce the electricity costs.

Inspired from Quanto [9] we would like to build a system that breaks down energy consumption into its constituents and also into logical activities. Quanto profiles energy at a fine grain level for embedded network device and also provides a framework for presenting the logical consumption of energy. Imagine a system which could give you the total consumption of energy spent on logical activities, for example playing video games or watching movies. This can give users an insight into their energy needs, also looking at this data from another perspective we can determine user behaviour from the the energy consumption data in the house. As a hypothetical situation, we can use the energy data to even track the frequent involuntary trips to the fridge for food or drinks. The first step to build such a system is to have the disaggregate the load information. In this report we present our prototype "Entracker", Energy tracker for homes which attempts at disaggregating the total power consumed in homes.

Our approach is based on concepts borrowed from Flick of a switch [16] and Electrisense [11] where the primary focus is on event detection of devices states. We try to extend these ideas and couple the energy breakdown techniques used in Quanto to build a system that provide the user with individual power consumption for any device. In this report we review the related work in this area and then describe our theory of operation. We describe the details of our prototype and implementation. In section 5 we discuss the evaluation schemes and analyze the results. Finally, we discuss the current limitations of the system and suggest ideas for future improvements.

2 Problem Statement

Every house has electrical appliances and their individual power consumption over time in unknown. We would like to build a system that can provide real time estimate of the power breakdown per device in a house given a power meter that records the total power consumption of the entire house and a device that determines the binary state of devices in the house. We state our estimation problem as follows, "To determine the binary states of individual devices and use this information to estimate the average energy consumed by each device in the house"

3 Related Work

3.1 Power Meters

As mentioned earlier, a few commercial smart power meters attempt to address the same problem. These power meters are more useful in analyzing patterns in energy usage and predicting future energy consumptions. They lack the granularity to measure or present to the user individual power consumption of devices.

Google Powermeter [3], EnergyHub [2], Wattsup Pro [8], TED 5000 [6] are smart power meters that are available in the market, they promote their products as an energy monitoring tool that helps user identify usage patterns. Where as GreenBox [4], Tendril [7] try to promote the case for smart grids where they target consumer empowerment to control energy usage and manageability at the user and grid level. Plotwatt [5] address the exact question we try to answer, but it relies just on the power data to calculate the individual power consumption of a device. They deploy survey based method or data from previous datasets to infer the breakdown and this approach may not be accurate as the devices themselves are not monitored.

3.2 Distributed Sensing

Distributed sensing methods deploy a sensor at every device which individually monitors the power consumption of device and then talks to a central authority or uploads the data to a server which processes this data to give analytics on the aggregate power and individual power breakdown over time. But installation of these sensors can be difficult and time consuming. Also we may need as many sensors as the number of devices we would like to monitor. Finally cost also is a factor when it comes to these sensors. Xiaofan Jiang et al. [13] have built “Acme” a wireless sensor and actuator network which monitors energy usage. iControl is another such commercially available energy monitoring solution using distributed sensors.

3.3 Single Point Sensing

The ease of installation and theoretical cost effectiveness have attracted people to build single-point sensing devices. The non intrusiveness of this technique also makes it a desired approach. Entracker extends the ideas presented in Patel et al. [16] which uses transient noises to detect devices and Gupta et al. [11] which uses the continuous noise generated by devices to provide single point device sensing and power breakdown. Transient noise is generated by most devices by the high surge in current when the device is actuated. Most devices also emit harmonics in higher frequency ranges which is termed as continuous noise. Both these work only deal with detection of electrical events. We combine these techniques to determine the actuation of devices and then record the total power consumption at periodic time intervals using a smart power meter and disaggregate the total energy into individual device consumption. In a follow up article in Smart Grid IEEE magazine [10], they try to address the same problem as us, but fail to give any metrics or evaluation schemes that help us assess their system. They give us a good understanding of how difficult this problem is due to the fact that obtaining ground truth is a non trivial task. There also have been other systems which have tried to answer the same question like the one developed by Roberts and Kuhns [17] which describes a non-intrusive appliance load monitoring(NIALM) to identify the largest energy usage appliance in homes. Jung and Savvides [14] describe their model which estimates the power breakdown on a large scale, like in buildings using the knowledge of device on and off states. This is a theoretical result and also their reliance on device on and off requires installation of binary sensors in the buildings.

4 Entracker Details

4.1 Hardware and Setup

In this section we describe the hardware setup of our data collection device . We emulated a house setting by creating a model house which consisted of various plug points and switches for devices like CFL lamp and bulb as show in the Figure 7. The model house was created in our systems lab and to isolate the noise from other machines and heavy duty electrical devices we installed a noise isolation filter(Tripp Lite IS500). We then connected the power meter (WattsUp Pro) in between the house electrical entry point and the

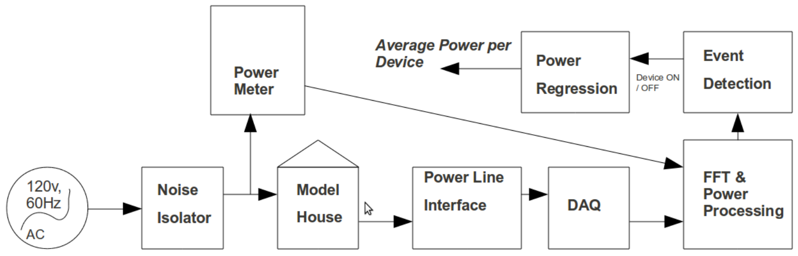
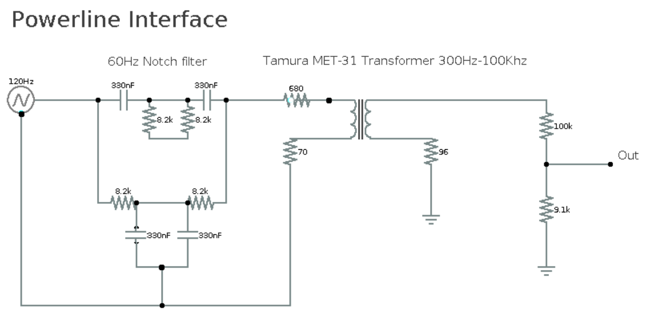
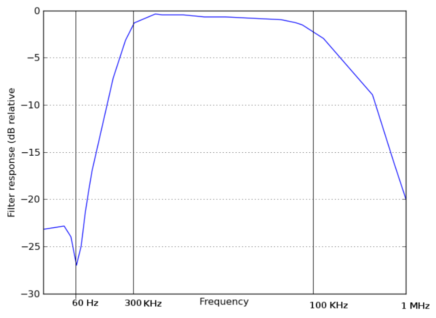


Figure 1: Schematic block diagram of Entracker prototype

noise isolation filter. The noise signals that the devices generate are recorded using a USB oscilloscope (NI 6128), although any USB oscilloscope could be used. Since we used a 60 Hz power supply the amplitude of the noise signal at 60 Hz would be really high and would mask the noise, hence we used a power line interface circuit similar to that mentioned in Patel et al. This power line interface consists of a notch filter at 60 Hz before passing through a band pass filter with 300 Hz to 100 KHz as passband. The USB oscilloscope samples at 200 Kilo samples/sec. The data from the oscilloscope is recorded using National Instruments Signal Express software, which dumps the recorded data into a text format.



(a) Powerline Interface Circuit



(b) Frequency Response

Figure 2: (a) This is the schematic of the power line interface circuit, (b) This is the frequency response curve for the powerline circuit which essentially is a band pass filter from 300 Hz to 100 KHz. Note the dip 60Hz due to the notch filter.

4.2 Software Details

We use NI Signal Express to record the data from the DAQ. It dumps the digital data into a text file which is processed by our software. The power data is also dumped into a text file periodically by the Wattsup Pro software. The reference times in these instruments are not the same and hence we wrote a module to synchronize the data we record from these independent sources. The DAQ samples at 200 kilo samples/s and the power meter samples 1 sample/s. We buffer the data for 60 seconds and then process the data for that time interval. Although we could do the processing in real time we prefer buffering of the data for optimizations where we look forward in the data and power data to better predict the device actuation's. The system prototype detects simultaneous events which are at least one second apart, this constraint is due to the lack of granularity in the power meter. The Entracker prototype application is written in Python which is known for rich support in data processing. Also python was an attractive choice as we used scipy, numpy and matplotlib libraries which gave us similar processing tools like matlab for FFT, graph generation and other advanced scientific operations. More details on scipy can be found on the website www.scipy.org and moreover its free.

4.3 System and Theory of Operation

Our system contains two modules: the signal processing unit and the power meter. The signal processing unit (see Figure 1) is DAQ which records the analog noise signal through a power line interface circuit, which in turn is plugged into any electrical socket. The hardware details are explained in the hardware setup section. The power meter should be connected at the electrical point where you need to disaggregate the load. Ideally this would be connected at the point where power line enters a house to disaggregate the load of the entire house. Each device has a characteristic noise pattern which is produced when it turns on, the systems collects these characteristics and uses machine learning to predict later when these devices are switched again. As mentioned earlier this approach borrows concepts used by Patel et al. where each device is classified based on its transient noise features when the device is stitched and also from Gupta et al. where each device is classified based on their continuous noise characteristics.

Based on results from Marubayashi [15] we understand that every electrical device produces a broadband noise when it is switched on. These can be either transient or continuous. Due to fast switching of currents when a device is turned on or off, a short lived noise pattern is injected into the electrical line which is termed as transient noise. After being switched on the devices produce periodic noise which can be termed as continuous noise. The continuous noise tends to exists until the device is switched off. This type of noise is a characteristic of devices using switched mode power supplies, which are prevalent in households today.

4.3.1 Detection of Transient Noises

Our first step is to detect the transient noise in the electrical line. Signal data is sampled by the DAQ and our software module converts this signal from time domain to frequency domain by performing a real FFT transformation. To detect these transient pulses we look at the changes in the signal noise by comparing successive FFT vectors. If the change exceeds a pre-defined threshold we isolate this as a candidate for further processing. This is done by computing the Euclidean distance between the successive FFT vectors. The threshold for detecting this was fixed statically and this was determined by experimental runs. We acquire 0.1 second sample of the signal data and the FFT is performed on this data slice. The FFT transformation results in a vector which contains amplitudes for frequencies ranging from 0 to 100 KHz. Each component in the vector corresponding to a frequency range and each of these bins correspond to 10Hz resolution. Random noise was observed in the frequency range 0 to 5 KHz, which may be attributed to attenuated noise beyond the filter circuit's band pass range (100 KHz). We disregard this frequency band for further processing.

These isolated candidates have distinguishable magnitudes for every device. This is clearly shown in the Figure 3, where we observe characteristic magnitudes for each device. It was further observed that over several instances of device actuation's the euclidean transient value was stable. We later use this magnitude which we term as "Euclidean Transient“. Magnitudes of these Euclidean transients for every device was averaged over instance and this was later used to classify the transients. We noted that the Euclidean transient was absent or was undetectable for off transition for devices. Hence we employ the above technique only for detecting the "switching on" of devices.

There are potential weaknesses with the method described above. Euclidean transients over a large number of instances may vary. Also certain devices have the Euclidean transient values in the same proximity and hence it affect classification. We can use more powerful methods, such as using an SVM classifier to

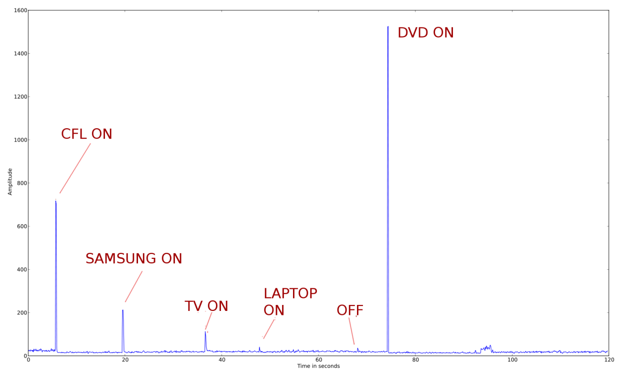


Figure 3: Plot of euclidean distances of the signal data over time. The transient peaks are seen when the devices are turned on.

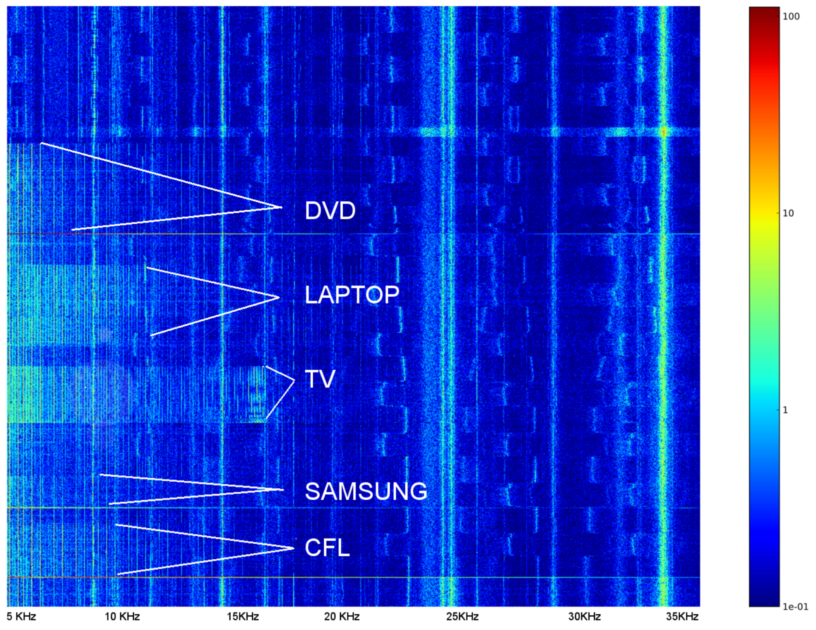


Figure 4: Spectrogram showing the continuous noise signatures when each of the device is turned on. Colorbar provided indicates the amplitude at each frequency. Y axis shows time

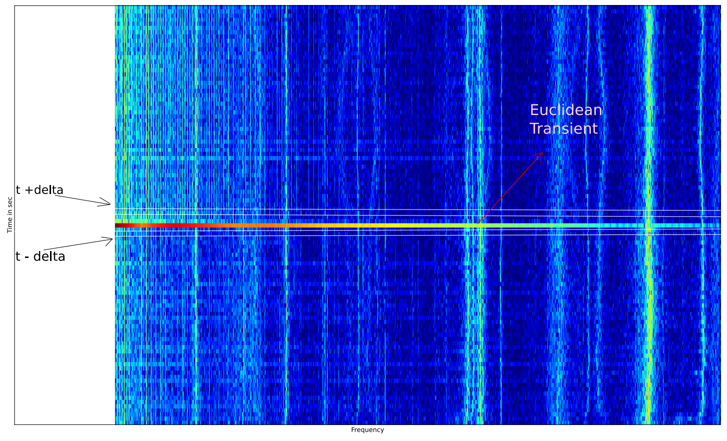
distinguish the spectral signatures of the transients, but we found that for our datasets just looking at the magnitude of the changes, i.e., the Euclidean distance, worked well.

4.3.2 Continuous Noise

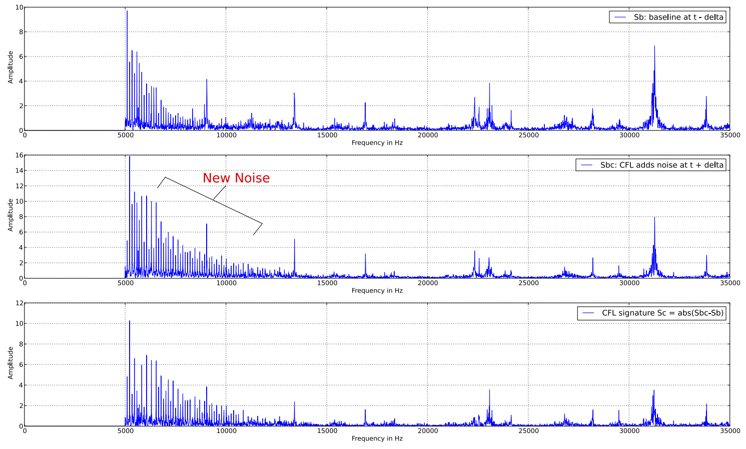
Devices that rely on switched mode power supplies generate electromagnetic interference which gets added in to the electrical line. This has a highly repeatable pattern in the frequency domain which can be isolated for devices and used for classification. The continuous noise persist until the device is switched off. Every time a device switches on we observe a transient noise for a brief period and then they emit continuous noise as show in the Figure 4. To capture the continuous noise signature of a device we first detect a Euclidean transient (see Section 4.3.1). In the Next step mark we the FFT vectors say S_{bc} at time t + δ which now has the added device continuous noise and the baseline vector say S_b (which is the vector at t - δ). Finally the noise signature obtained by taking the difference of the two vectors S_{bc} - S_b is saved as the continuous signature for the device. The Euclidean transients aid the process of isolating the continuous noise signatures and avoid the SVM classification to be run at every time slice.

The process of capturing the continuous noise signature for CFL is illustrated in the Figure 5(a). The

Euclidean transient for CFL is clearly visible in the figure which serves as our reference point to obtain the baseline vector and the vector with added noise due to actuation of the CFL. In the Figure 5(b), we show plots which show the FFT vectors at time t - δ, t + δ and also the signature vector which is the absolute difference of these vectors (we used δ = 0.2sec). Manual inspection of the noise spectrum of these continuous noise signatures of devices led us to believe that the dominant differences in various signatures were mostly visible in the frequency range 5 KHz to 35 KHz. Trial runs with the entire frequency spectrum produced similar results, hence we used the smaller frequency range for faster processing. We believe that the frequency range beyond 100 KHz could contain rich features as shown in the work by Gupta et al.



(a) CFL Continuous Noise



(b) CFL Signature

Figure 5: (a) New noise is observed when CFL switches on, we show how we capture the continuous signature using the transient signature as a reference. (b) Shows the signature profile for baseline, baseline + CFL , computed CFL signature.

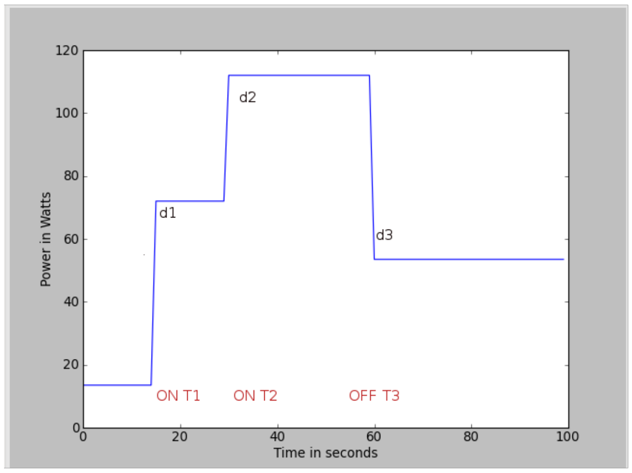


Figure 6: The power change when devices get turned on. d1, d2 represent the power change when TV turns on and d2 represent the change when laptop turns on. d3 is the change when TV turns off, note that the d1 is approximately equal to d3, hence we can attribute the change in power to TV in this example.

4.3.3 Detection of Off Transitions

The above technique worked well for classifying the device on states, but did not work as well for off states. The main reason being that the euclidean distances were not visible for the off transients . This meant that we could not get the potential signature to be marked for classification of the device. Through preliminary analysis we found that the classification was accurate for on transitions in all datasets up to 80 percent where the dataset contained around 60 transitions. Hence we could already attribute the power change to the device which we determined to be on. As seen in the Figure 6, When a device is turned on we measure the average power change since the device turned on and store this power change as the power transient for the device that caused it. When there is a change in power consumption we try to detect which device was turned off. To do this we note the change in power when a device was turned off. Based on the proximity to the power transients that we stored previously, we mark that device to be turned off. The proximity is determined by calculating the difference between the noted power change and the stored power changes for each device.

4.4 Learning Phase and Classification

We switch on every device in our system for a few times and the continuous signatures using the technique described in the above section, then we label them manually to build our training set. To classify these continuous signatures we employ support vector machines (SVM). The final classification of the on transition is determined by a combined technique which uses the magnitude of the Euclidean transient and SVM output to determine which device was the most likely to be switched. Each of the individual techniques return an order of rankings. To avoid static attribution of weights to either techniques, we combine the two classification ranks using a technique similar to Borda count [1]. This technique determines the winner, or in our case the most likely device, by combining the rankings in the order of preference by each method. A score is given to each device in each of the individual technique by determining the fraction by which the device was more likely to be the best choice. Essentially the scores in the two domains are normalized, so as to combine them into a single score. This single score is then used to determine which device was the most likely to be switched.

SVMs perform classification by constructing a N-dimensional hyperplane that separates the data based on the learning data into multiple sets. We employ a technique called binary SVM classification where we construct the SVM model using the learning data as follows. Consider a signature set which contains signatures for all the devices in the system, The model for device A would be constructed by labelling all signatures for device A as ”device A” and all other signatures in the collection as “not device A”.Once these models are created for each device we can use them to classify a given potential signature. We implement this binary SVM classification using SVMlight [12] which is a freely available SVM implementation. SVMs are a desirable tool because they provide a large feature space compared to the potential training set. Since our code was written in python, we used pysvmlight which is a python binding for the svmlight, also downloadable freely from the website.

4.5 Power/Load Disaggregation

The main objective of Entracker is to disaggregate the load at any given time and attribute them to individual devices. The device consumption may not be constant, but the variation for any given device may be hard to calculate. To avoid a static calculation of this variation, we use a regression model similar to least squares regression that is used in Fonseca et al. [9] to attribute the energy usage to devices. Every 60 seconds, we create a vector containing information about device states. This is done by maintaining a global system state which stores a binary value indicating if a device is on or off at any given time. We then correlate these device states to the power data we measure from the power meter. We also record the power consumption of the entire system every 1 second, as the Wattsup Pro meter does not allow to record the power data below this interval. For each interval of 60 seconds we have a set of 60 vectors which contain the what the system state was at every second and also another vector which contains the total power consumption of the system at every second. This data corresponds to a system of equations as shown below

$$\underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_{60} \end{bmatrix}}_{\mathcal{P}} = \underbrace{\begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1n} \\ d_{21} & d_{22} & d_{13} & \dots & d_{2n} \\ d_{31} & d_{32} & d_{13} & \dots & d_{3n} \\ \vdots & & & & \\ d_{60,1} & d_{60,2} & d_{60,3} & \dots & d_{60,n} \end{bmatrix}}_{\mathcal{D}} \times \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}}_{\mathcal{X}}$$

P contains the total power consumption recorded by the power meter every second in the recorded interval, D contains the state of devices in the system at every second, each state takes binary values and X represents the variable to be solved which is the average device consumption. These system of equations is modelled as a minimization problem where our objective function is to minimize $||\mathcal{D} \times \mathcal{X} - \mathcal{P}||_2$, essentially we are trying to reduce the least squares error. The solution to this minimization problem gives the average power consumption of each device for that time interval.

4.6 Deployment

The system was setup as shown in the Figure 7. We had a noise isolation filter which isolated the noise from the building and we connected the power meter between the noise isolator and the model house. The devices were plugged into the outlets provided in the house. The noise generated in the house was sampled by the data acquisition unit (NI 6128) through the powerline filter circuit connected to one of the outlets. To generate the training data and capture the device signatures we simply switched on each of the device one by one for 5 times and then collected their Euclidean transients and continuous noise signatures as described in the section 4.3.1. The training phase is fast as we are not concerned about the power state of other devices. Hence this allows us to easily add more devices in the system. The installation of Entracker would be really easy in residences as there is no pre-requisite state for the house to be in. Just as a side node we would like to mention that we noticed a baseline noise (when no device was turned on) in our system, which happens to be common for a lot of devices and in addition there was a baseline power consumption even when any of

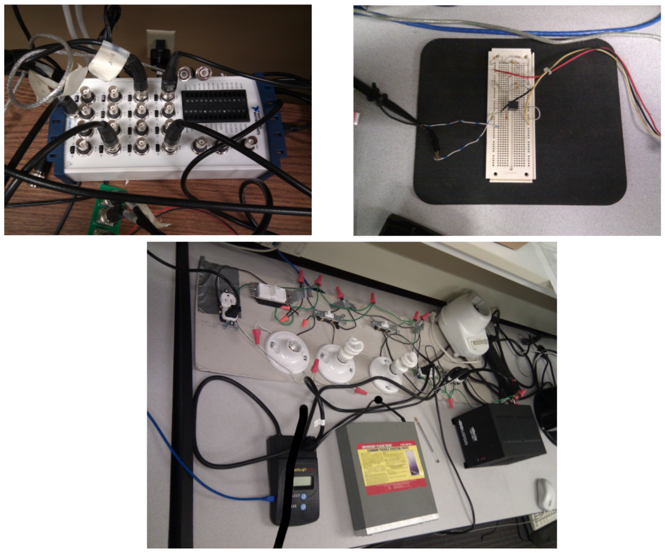


Figure 7: NI 6128 DAQ (Top Left), Powerline Interface Circuit (Top Right) and Model House (below)

the devices were not turned on. We refer to this as phantom load or baseline load and found it to be around 13.5 Watts.

5 Evaluation

Similar to Quanto, Entracker tries to answer the question *where have the joules gone?* Evaluation of energy disagrregation is a major challenge. It is a difficult task to record the ground truth indicating when a device was switched on, moreover the energy consumed by it fluctuates from the rated wattage on these devices. So our main motivation behind evaluation is how often can we detect the correct transition and to evaluate the breakdown of energy consumption per device. In our evaluation we try to answer these two questions:

1 How often can we detect the device on and off states correctly?

2 How accurately can we attribute the total power consumption at any given time to individual devices?

To simulate the electrical events in a house and to explore all the possible device states, we perform a random walk on all the device states. For each dataset, we start the system state from the initial state where all the devices are turned off. We then generate a random number between 1 and the number of devices and then flip the state of that device. This random walk allows us to visit all the corners of the hypercube that represents the device states in the system. Each of the datasets we report below are a trace of the system over 30 minutes of state changes. Since the events are randomly generated we have fairly equal number of transitions for each device. This helps to test detection of devices when other devices are on and also tests the variation of power breakdown when the device is actuated at various intervals. We used the training data which was recorded a few weeks before the actual evaluation using these datasets.

To answer the first question as to how often can we detect the power states of devices correctly, we processed the trace collected from the dataset. We found that Entracker was able to classify around 70% of the device actuation's correctly. We had a higher accuracy of around 80% for detecting the on transitions. The accuracy is lower for all events including off transitions because we pay double penalty for every wrong on classification we make. Every misclassification of on transition results in misclassification of the off transition too. It was also observed that we missed a few transitions due to the fact that we had a static

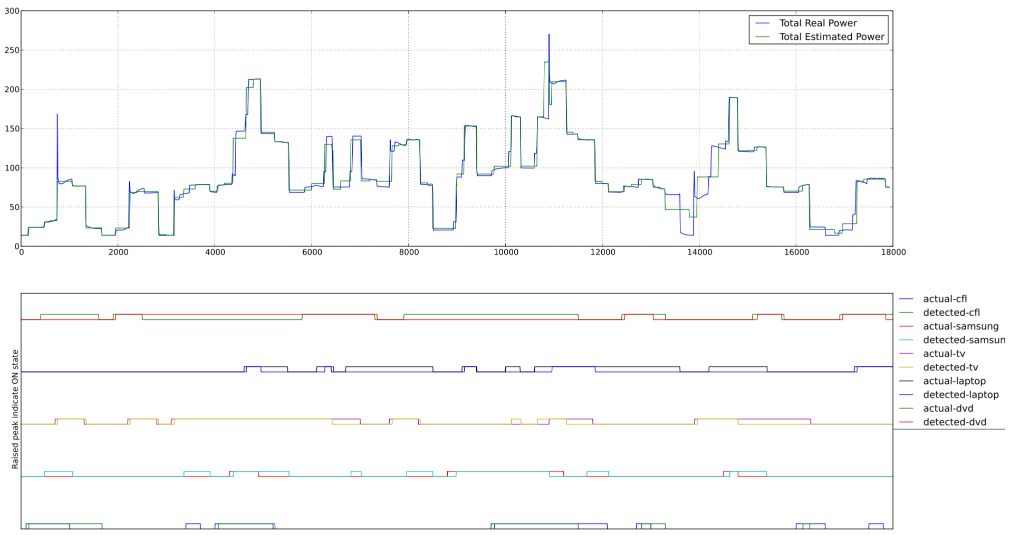


Figure 8: A trace from one of the datasets. The subplot above shows the fraction of total power we were able to breakdown. The one below shows the actual device on/off state versus the estimated device on/off state. Note that the actual and estimate could have a small lag due to human errors while device actuation.

threshold for detecting the Euclidean transients. The accuracy of our method is less than that achieved by Patel et al. [16] and Gupta et al. [11] This may be attributed the sampling rate of our DAQ, which greatly reduced our feature space. As these previous works indicate they observed significant presence of continuous noise beyond 100 KHz.

Now given the device on and off states we need to breakdown the total energy consumption into its constituents. We first evaluate based on the fraction of the total power could we could account for. This is graphically shown in the Figure 8 where we plot the total power consumed and the estimated total power at any given instance. Figure10 show the power consumption of individual devices in each time slice, the each of the colored area in the graph would give the total power consumed by that particular device. For all the data sets it was observed that we could account for over 95% of the total power consumed. The next step is to determine how accurately we attribute the power once we estimate the device states and have the recorded power data. There is no actual metric that we can use to evaluate this because the ground truth of what each device consumed is unknown. Thus we base our accuracy on the rated wattage for each of the device and the average power we observed over time in all our datasets. This is shown graphically in the 11, we can observe here that when a device is misclassified, its power attribution deviates from the average. This graph gives us an indication of the variation in the estimated average power of each device over time.

The average power consumption estimated for each device in datasets is show in the Table 2. The average power for each device was close to its rated wattage except the samsung lcd monitor. The reason for this can be attributed to mis-prediction in the system, which propagates and the penalty for it is substantial. Accuracy of on events were close to 80% in all datasets and accuracy of detection of power states were close to 70% (Table 1). This advocates for the need to employ correction mechanisms to lower the impact of mis-predictions or even eliminate them. If we have sufficient information of a particular device for a long period of time, then deviation of the estimated power of a device from its average power consumption could be useful information to identify a mis-classification. The regression output can be used in predicting

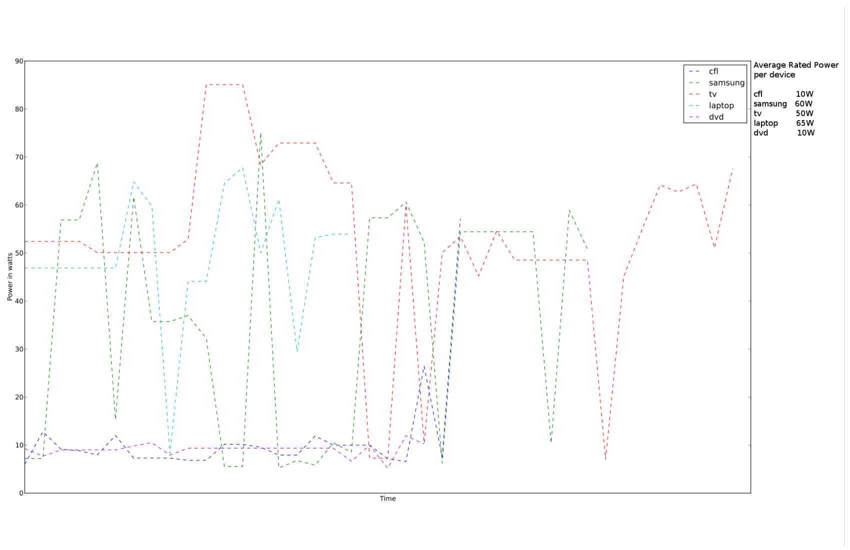


Figure 9: Variation of the estimated power for each device by Entracker over time. Note that when devices are mispredicted they can be attributed no power or larger share of the power by Entracker.

when the device turns on or off the next time, resulting in a constant feedback into the system to indicate the mis-classification of a device or even detect a missed transition. We leave these ideas to be explored as future work.

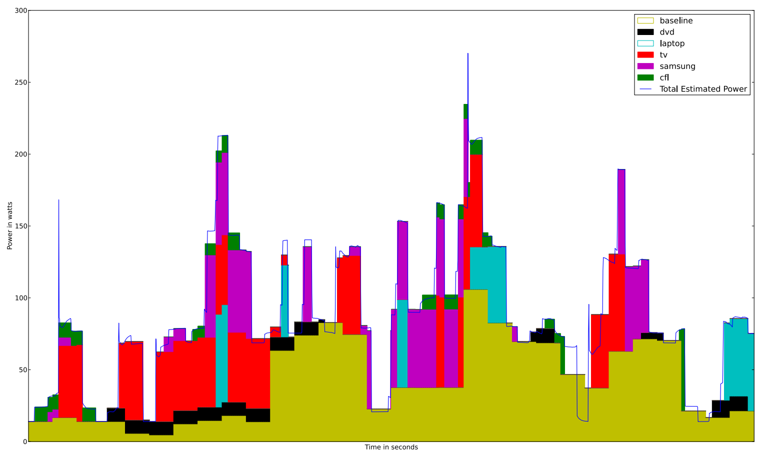


Figure 10: This figure shows the constituent power consumed by each device over time. The recorded actual power is also indicated.

6 Current Limitations and Future Work

Use of continuous noise in our classification scheme limits our approach to classifying switched mode devices. Resistive and mechanically switched loads that generate only transient noise (Incandescent Bulb, Toaster, Fan etc) can not be classified. But since we also look at transient noise features, we could adapt

Table 1: Summary of results: Shows the correct % of device transitions detected

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
OFF	63	48	56	58	54
ON	77	77	81	72	79
Total Accuracy	72	63	67	67	68
Total Power Accounted for	96.3	95.6	95.2	94.5	96

Table 2: Shows the computed average power of each device (in Watts) obtained from taking the average of regression results in each dataset

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Rated Wattage
CFL	10.15	13.74	11.38	14.48	20.5	10
Samsung	39.54	38.69	36.36	38.93	36.63	60
TV	48.9	50.76	53.31	58.03	45.15	50
Laptop	52.8	54.9	49.28	56.43	54.97	65
Dvd	9.62	9.15	9.09	8.52	10.32	10

our system to detect such devices too. Results indicate that we had a less accurate technique to detect the off transitions. This could probably be improved by using the regression information to classify the off transients. Another technique that could be used is to associate a confidence level every time we classify a event and override a previous classification if we detect the same transition again in a small window. We did not test devices that had multiple power states, for example we noticed that laptop consumes different amounts of power depending upon the charge in the battery, It consumed half the power when the battery was charged at 90% completion. Hence in our datasets we made sure that the battery was never charged beyond this point. Certain heavy voltage devices produce various transients and could be a potential issue for the system to handle these transients. We believe that instead of analyzing a big slice of the frequency spectrum, it is advantageous and efficient to extract dominant features from the device signature and use these features to classify a potential signature.

Currently we only analyze certain frequencies and their amplitudes for classification but disregard the phase component. The noise signatures for the same device could vary depending upon the phase of all the components, not just that of the device but also the other devices. We tried to incorporate phase into our model and failed at our attempt to do so. But we still firmly believe that accounting for phase during the classification can boost the performance of the classifier. Although the flip side is that accounting for the phase in the learning model is far from trivial. Finally we believe that once we have a reliable event detection and power estimation, we could start answering questions like how much energy was spent playing video games or Watching a movie?. We believe such logical tracing of data can give users better understanding of their energy usage data. Additionally we could get logical breakdown of power per room or floor, depending upon the granularity that the user desires. Also looking at this data from a different perspective, we believe power data could be used to infer user behavior.

7 Conclusions

In this report we presented a single point sensing system which is capable of disaggregating the total power consumption of a house into its constituents. This work lays the foundation for a lot of applications like home automation systems and smart grids, which leverage the individual power consumption information. Such automated system can enable the user to identify faulty device, their usage pattern and thus help them reduce the energy consumption. This helps the users to reduce the dollars they spend on electricity and also has an environment impact by reducing the carbon emissions. Our event detection scheme may not have been highly accurate as the previous works but we believe that the concepts in this report can be applied in conjunction with similar methods and can produce highly accurate estimation of average power of individual devices. In spite of having an inferior event detection scheme we still had high accuracy on the average device power consumption, as suggested by our evaluation.

References

[1] Borda count. `http://en.wikipedia.org/wiki/Borda_count/`.

[2] Energyhub. `www.energyhub.net`.

[3] Google powermeter. `http://www.google.com/powermeter/about/`.

[4] Greenbox. `http://www.google.com/powermeter/about/`.

[5] Plot watt. `http://www.plotwatt.com/`.

[6] Ted5000. `http://www.theenergydetective.com/store/ted-5000`.

[7] Tendril. `http://www.tendrilinc.com/`.

[8] Wattsup pro. `http://www.wattsupmeters.com/`.

[9] Rodrigo Fonseca, Prabal Dutta, Philip Levis, and Ion Stoica. Quanto: Tracking energy in networked embedded systems. August 31 2009.

[10] J Froehlich, E Larson, S Gupta, G Cohn, M Reynolds, and S. Patel. Disaggregated end-use energy sensing for the smart grid. Pervasive Computing, IEEE, 2011.

[11] Sidhant Gupta, Matthew S. Reynolds, and Shwetak N. Patel. Electrisense: single-point sensing using EMI for electrical event detection and classification in the home. In Jakob E. Bardram, Marc Langheinrich, Khai N. Truong, and Paddy Nixon, editors, *UbiComp*, ACM International Conference Proceeding Series, pages 139–148. ACM, 2010.

[12] Joachims. Svmlight. `http://svmlight.joachims.org/`.

[13] Deokwoo Jung and Andreas Savvides. Estimating building consumption breakdowns using ON/OFF state sensing and incremental sub-meter deployment. In Jan Beutel, Deepak Ganesan, and Jack A. Stankovic, editors, *SenSys*, pages 225–238. ACM, 2010.

[14] Deokwoo Jung and Andreas Savvides. Estimating building consumption breakdowns using on/off state sensing and incremental sub-meter deployment. Sensys, 2010.

[15] G. Marubayashi. Noise measurements of the residential power line. pages 104–108. International Symposium on Power Line Communications and Its Applications, 1997.

[16] Shwetak N. Patel, Thomas Robertson, Julie A. Kientz, Matthew S. Reynolds, and Gregory D. Abowd. At the flick of a switch: Detecting and classifying unique electrical events on the residential power line (nominated for the best paper award). In John Krumm, Gregory D. Abowd, Aruna Seneviratne, and Thomas Strang, editors, *UbiComp*, volume 4717 of *Lecture Notes in Computer Science*, pages 271–288. Springer, 2007.

[17] M .Roberts and H.Kuhns. Towards bridging gap between the smart grid and smart energy consumption. 2010 ACEEE Summer Study on Energy efficient in buildings, 2010.

8 Appendix

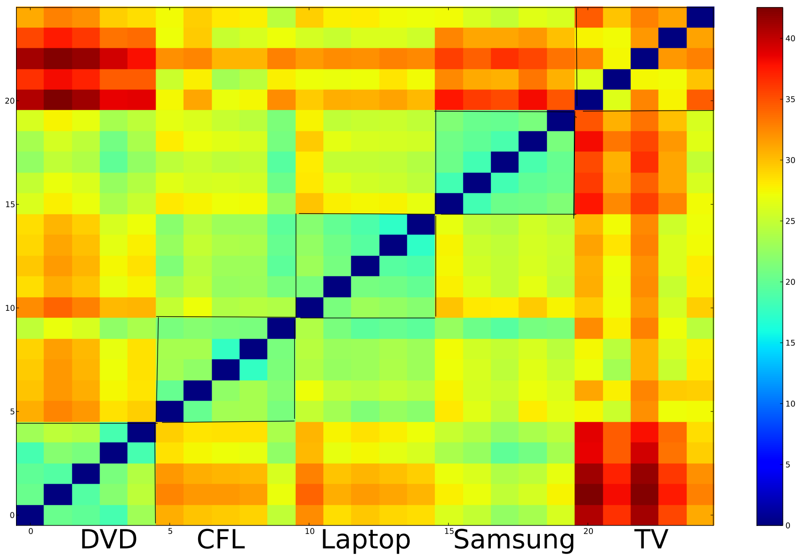


Figure 11: Device Correlation matrix. This matrix plots the Euclidean distance between the device signatures of various devices. For each collection of device signatures we can see that the difference between them is low. Significant difference is observed between signatures of various devices.