Internet monitoring of a load and price trends optimization of the electricity grid

Thomas Lataille, Guillaume Martin, Matthew McBrien
21th April 2019

***Abstract*— As the grid modernizes and enters the internet era, customers will be able to view, understand, and control their power usage in a faster and more real-time manner than was originally envisioned when designing the legacy grid. Customers will be able to understand how their electricity is being used with much more detail. While knowledge of their electrical use is good, it is vital that this new technology comes with the ability for consumers to also control their usage for optimal efficiency and cost-savings. As discussed in [1], the grid and the internet are converging quickly as the amount of data available from the grid explodes and the internet is the perfect place for control and monitoring of such data. In this paper we discuss our results from creating a proof-of-concept system for controlling a small DC motor over the internet using a Raspberry Pi and managing its power usage by optimizing against a Peak Time Rebate.**

I. Introduction

In this paper we discuss a proof-of-concept system for monitoring small loads (a DC motor) over the internet using a Raspberry Pi as a three-in-one device for controlling the motor, recording the current and voltage statistics, and hosting a web server so that a client can use a custom-built application to access and control the load. Additionally, the client application is integrated with data collected from a sample [2] of 5,567 London Households who took part in the UK Power Networks Low Carbon London project between November 2011 and February 2014. This allows us to create a hypothetical demand curve. In our scenario, we imagine an upcoming day when a rebate is offered for several hours. By using the load data, we can create an optimal load curve considering the rebate. The client application can then remotely control the load to match the load curve generated with the rebate leading to maximum savings for the customer.

Similar work has been done in internet monitoring and control of loads. For example, in [3], the power supplied to a load was tracked and could be disconnected in the case of surges. The data was also automatically posted to a website. And in [4], a proof-of-concept system was developed for client access and control of data from a smart meter using a ZigBee network. In [5] an IoT system was developed to integrate with the smart grid allowing for real time monitoring. In this paper we discuss our work and results from implementing real-time monitoring and control of a load as well as automatically controlling the device for peak savings for the customer.

II. Our system and methodology

Our system is composed of 3 different layers:

- The device layer where the load is electrically controlled;
- The cyber layer where information is sent on internet and highly protected using specific protocols;
- The client layer where the customer decides to turn on or off his load;
- The market layer where the real time pricing and forecasting are made in order to enable the peak time rebate algorithm.

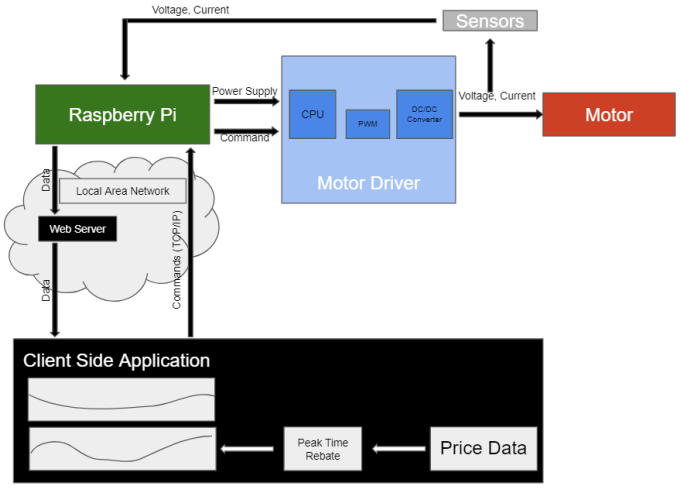

Figure 1: Original design scheme of our system

A. The device layer: control of the load

The electric load is modelized with an electric motor. The DC motor used is a simple DC Gearbox motor with an input range of 3 to 6 VDC for a maximum speed of 200 RPM. The equivalent circuit for a DC Motor is the following :

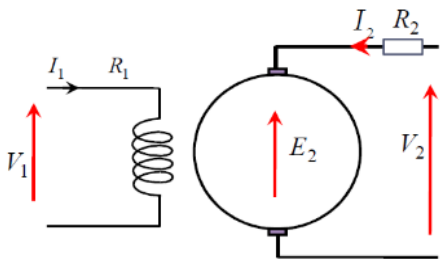

Figure 2: Equivalent circuit of the electric motor.

Where the left side is the inductor, the right side is the armature, Γ is the torque and Ω is the speed.

The corresponding equations for a motor behavior are:

$$\begin{cases} V_1 = R_1 I_1 \\ V_2 = R_2 I_2 + E_2 \\ \Gamma = M_1 I_1 I_2 \\ E_2 = M\Omega I_1 \end{cases}$$

Assuming we are using a permanent magnet DC motor, when we increase the voltage V2, then we increase the speed of the motor. And when we increase the current I2, we then increase the torque of the motor. The control of the motor is thus the control of the input voltage and current.

This can be achieved by using a DC/DC converter. For example, a buck-boost is a kind of chopper we can use to increase or decrease the output voltage when putting a constant DC voltage at the input.

The general idea is to take a H chopper, and to control the switches using a PWM signal (Pulse-Width Modulation).

To summarize, using a constant power supply, we can control the input voltage and current of the motor, using a DC/DC converter, to control the speed and the torque of the motor. The control signal will be the PWM signal we put into the DC/DC converter.

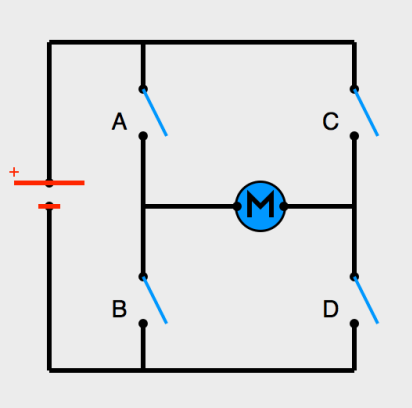

Figure 3: H Chopper converter.

The PWM acts as the signal that controls the opening and closing of the switches. By varying the duty cycle of the PWM, the speed of the motor can be changed.

For this project a small motor controller is used to control the electric motor.

The component is the TB6612 DC/Stepper Motor Driver [6] from Adafruit. It will be used to generate the right PWM signal to control the speed of the motor, and also to control the direction of rotation (either clockwise or counterclockwise) using another H chopper with the following conditions on the switches:

- When A and D are closed, and B and C are open, the motor is rotating clockwise.
- When A and D are open, and B and C are closed, the motor is rotating counterclockwise.

The set up for the control of the electric motor is as follows:

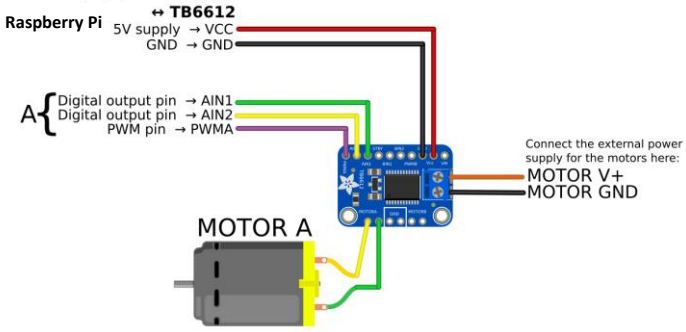

Figure 4: Schematic of the set up to control the electric motor [7].

A sensor is added and connected to the motor and to the Raspberry Pi to record the current and voltage sent to the motor. Therefore, the power consumed can be computed and the energy consumption of the motor can be recorded.

B. The cyber layer

Monitoring Power

Power was recorded using the Pi and an IC module called the INA219. The INA219 is capable of recording both current and voltage of a load and then publish this information using I2C. The INA219 can handle up to 26V and 3.2A, so it was easily capable of handling our much lower stress system.

Python modules are available online for reading from the INA219 device on the Pi, and these were used to load the data onto the Pi before submitting them to the server for publication. While the IC is capable of reading at a much faster rate, the Python code developed read in current and voltage values every 0.5 seconds and then posted this data to the web server.

Client Application

The client application that was created for this project acted as a mock-up of what a client might see and provided basic controls over the load. The application was made using Python, the GUI library Tkinter, and the plotting library matplotlib.

The client application included four charts (one dynamic and three static) and buttons that could control the speed of the motor. The speed of a DC motor is directly correlated to its power, so by increasing the speed of the motor, we could increase the power as well and vice versa.

The dynamic plot on the client application plotted all of the data available on the web server hosted on the Pi. After every second, the client application would use the server's API to ask for the current and voltage data. It would convert this to power using the equation. It would then update the plot with the newest data.

The three other charts were the prices of the electric load, as well as the peak time rebate forecasted loads and prices.

In addition to the charts, the client has the ability to control the speed of the motor using the client application. Two buttons, an increase and decrease button, are provided for changing the speed. Since the motor is run using a PWM output, an increase command would increase the duty cycle of the PWM output by two percentage points and a decrease command would do the opposite. Altering the PWM duty cycle has the effect of changing the power draw and the speed of the motor.

When the client application is started, there are two options for its use presented to the user. A user can choose to have the program automatically follow the new demand curve as calculated using the rebate information (discussed in a future section), or the user can have complete autonomy over the output of the motor and not follow the curve. If the user chooses to not follow the curve, then the application will plot all of the data available on the server and the motor will respond as expected from commands from the client application.

If the user chooses to follow the rebate curve, then the application takes control over the motor by sending increase and decrease commands independently of the user. When this option is selected, the application tries to simulate the results found in the new demand curve. This data represents an entire day's power data. To expedite this process for the purposes of demonstration, the application acts as if the new demand curve represents only two minutes of data, so each hour only takes five seconds in the simulation. Additionally, the new demand chart represents an entire households use of electricity so the values are too large to mimic with a single DC motor. To account for this, the maximum power output of the motor was determined and compared to the largest value of the new demand plot. This was then said to be 100% power output and the power output of every other timestamp was measured by comparing against the maximum value.

The application attempts to have the dynamic power plot match the static new demand plot by comparing the current power level to the power level at the corresponding hour of the new demand plot. If the current power is below the corresponding hour, then increase commands are sent until it matches as closely as possible, and the opposite is done when the power is too high. As can be seen from figure 15, the application closely matches the shape of the new demand plot.

C. The market layer: load capacity and price optimization

Having the possibility to choose the best price of electricity for your device in terms of the electricity market would give a huge edge to reduce your electricity bill. By connecting the dashboard of the electrical load to a forecasting data folder, consumers can see the trends and real time price they would want to pay.

In order to carry out cost optimization of a household, we needed 2 data sets:

- The electricity market price;
- The load consumption profile.

The data set used for the electricity market price was taken from an open source real time data folder [8] which includes hourly prices from the el. This publication includes recorded data from 10 days that act as our forecasted prices.

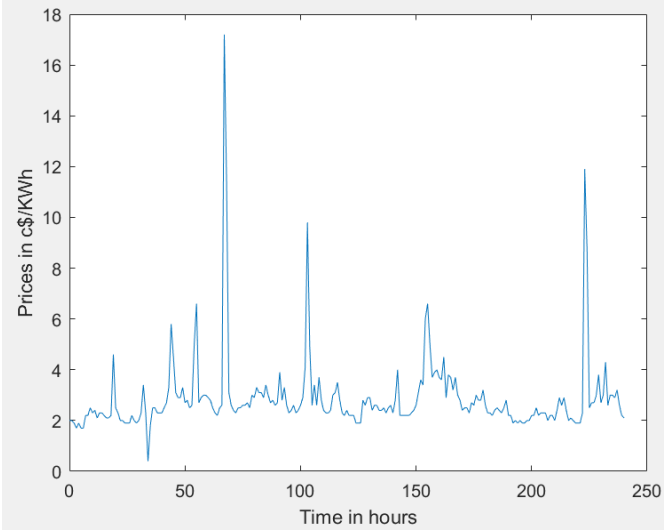

Figure 5: Prices of the electric load from the 03/30 midnight to the 04/08 midnight

The consumption load profile [2] was taken from a sample of 5,567 London Households who took part in the UK Power Networks Low Carbon London project between November 2011 and February 2014. We converted the half hour to an hourly consumption load profile through 3 days between the 30th of March and the 1th of April 2014.

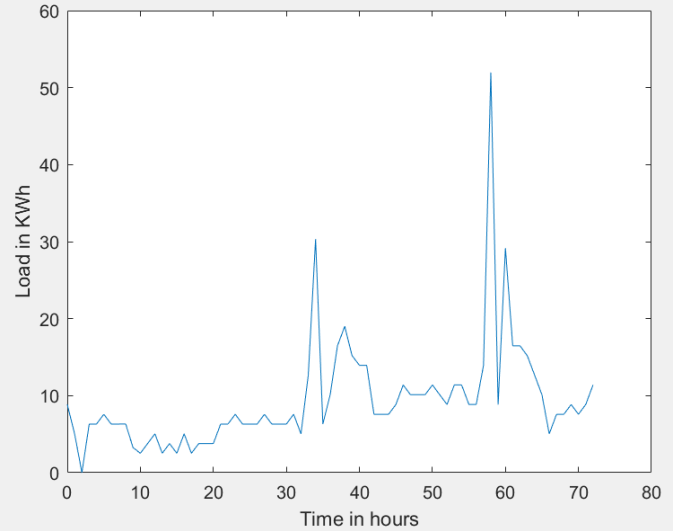

Figure 6: Electric household load consumption between the 03/30/2014 and the 04/01/2014

When looking at the load profile, we want to shift the high demand of the electrical consumption in order to prevent any saturation of the load. We will use the Peak Time Rebate optimizer by rewarding the customer to reduce its load consumption during peak hours.

In order to reduce their electricity bills and the load during peak hours, customers will receive a cash rebate for each KWh of load that they reduce below their baseline usage during the event hours. This optimization problem is called the peak time rebate or PTR. It requires the establishment of a baseline load

from which the reductions can be computed. Two possibilities are offered to the customers using our interface:

1. If customers respond to the Peak Time Rebate, they get cash rebate for each KWh of load that they reduce below their baseline usage during the event hours.
2. If customers don't respond to the PTR rates, their bill remains exactly the same as they would be under their existing tariff. This makes the program politically more attractive.

In order to find the best times for which the load must be reduced, we will run a demand shifting optimization algorithm. The main key is to minimize the shifted cost of load reduction, noting that the original demand must be equal to the sum of the demands shifted for all different times.

$$\min \sum_{i=1}^{T} \left[\sum_{j=1}^{T} (\pi_j - \pi_i^{REB} + \pi_j^{REB}) P_{i,j}^{shift} + \pi_D \sum_{j=1}^{T} |i-j| P_{i,j}^{shift} \right]$$

$$such\ that \sum_{j=1}^{T} P_{i,j}^{shift} = P_i^{orig}$$

$P_{i,j}^{shift}$: Demand shifted from time t=i to t=j

$C_{i,j}^{D}$: Cost of discomfort associated with $P_{i,j}^{shift}$

π_D : Price of discomfort

π^{REB} : Price of rebate

We carried out the peak time rebate algorithm for one day on the 1st of April. The idea behind our data analytics algorithm is to prevail peaks at certain hours of the day.

Thus, we chose to arbitrarily avoid a peak consumption of electricity at 10 am. We defined the rebate accordingly to redistribute the power at a different time of the day, as follows:

Hours of the day	12 am	1 am	2 am	3 am	4 am	5 am	6 am	7 am	8 am
Rebate	0	0	0	0	0	0	0	0	14
Hours of the day	9 am	10 am	11 am	12 pm	1 pm	2 pm	3 pm	4 pm	5 pm
Rebate	14	20	20	14	14	0	0	0	0
Hours of the day	6 pm	7 pm	8 pm	9 pm	10 pm	11 pm			
Rebate	0	0	0	0	0	0			

If the customer chooses to reduce its consumption at the peak time 10 am, then his consumption and his price rating would follow the curves below:

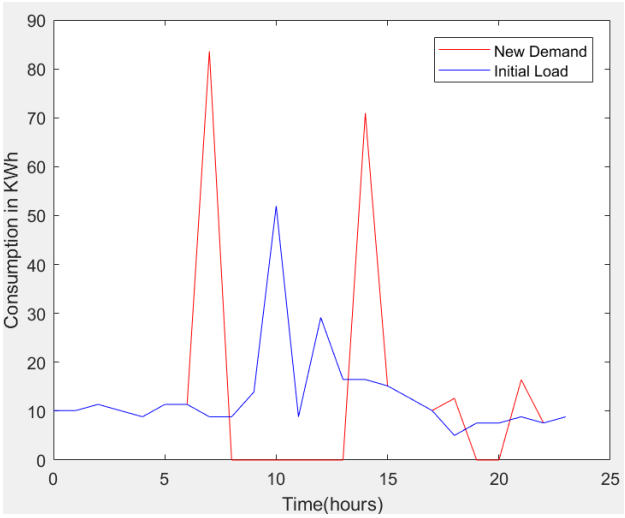

Figure 7: Initial load and new demand on the 1st of April when running the Peak Time Rebate algorithm

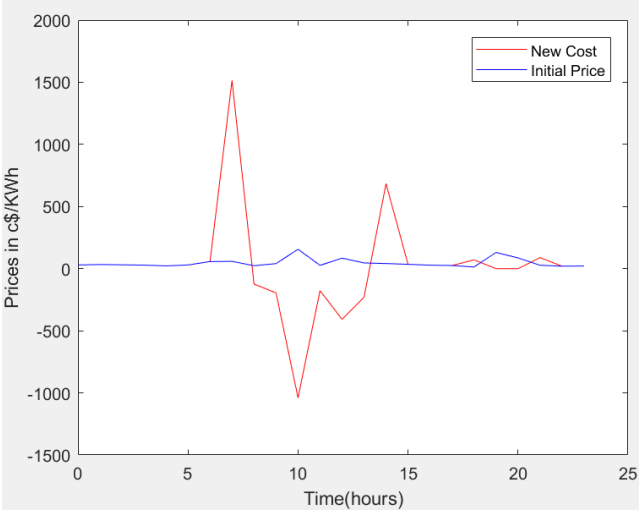

Figure 8: Initial Price and new cost when applying the Peak Time Rebate algorithm

The customer would gain $5.49 in one day when changing his consumption to the peak time rebate algorithm. This method is both a good incentive to help the customer reduce his consumption and help the grid to suffer less from high peaks of electricity by redistributing the load power consumption.

III. RESULTS

When combining the layers together, it results a system that control the speed of the electric motor remotely via a client interface accessible online, and that sends data to the Raspberry Pi through the internet.

The mock-up client interface displays the trends of the market in terms of the predicted load consumption per hour and the price per hour. It is also displayed a graph of the adapted load consumption that would allow the user to better distribute his energy consumption over the day, and by doing so, save money.

The final set-up is shown in the figure 9 and 10.

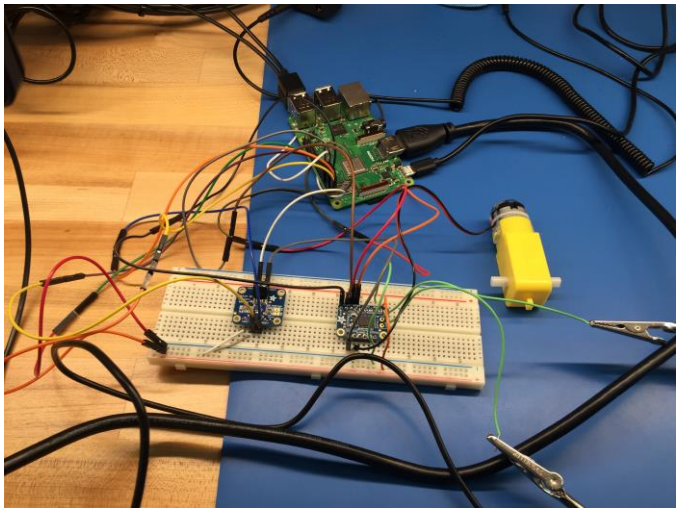

Figure 9: Picture of the set-up, electric circuit view.

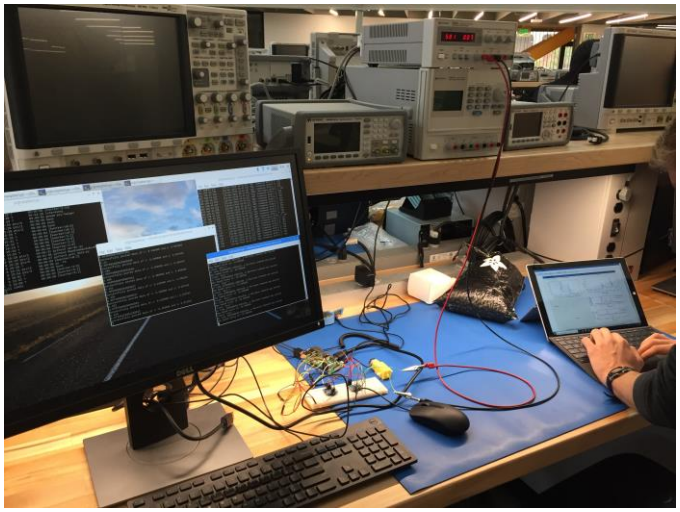

Figure 10: Picture of the set-up, global view of the system.

On the large screen, the Raspberry Pi operating system is displayed, as shown in figure 11. Four terminal windows are open. The one on the top left corner is used to type the commands, the middle one is used to read the data from the sensor, the one on the top right corner is used to read the connection status to the server, and the last one is used to follow the commands send to the Raspberry Pi.

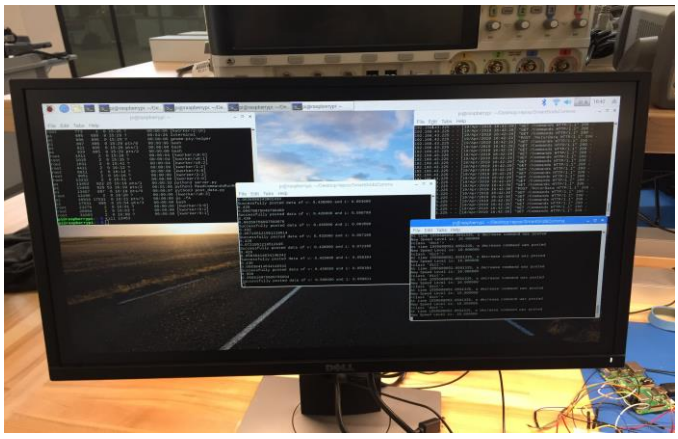

Figure 11: Picture of the screen of the Raspberry Pi.

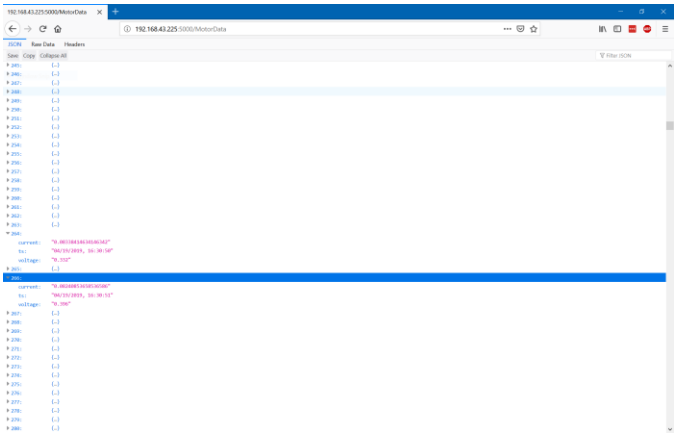

Figure 12 : Picture motor data page.

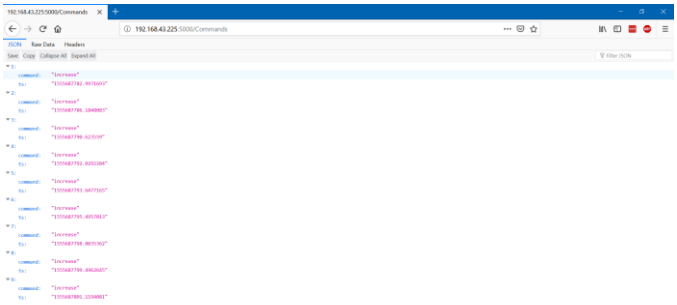

Figure 13 : Picture motor data page.

The motor data page and the command page, in figure 12 and 13, show the commands sent to the Raspberry Pi through the server, and the data from the sensor that are received from the server. These captures show the raw json formatted data which had to be interpreted by both the Pi and the client application.

A. Response to an increase or a decrease of the motor speed.

The algorithm developed allowed the client to increase or decrease the power output (via the motor's speed) by pressing a button on the client interface.

The power transmitted to the motor and monitored using the voltage and current from the sensor, is displayed on the client interface, as shown on the figure 14. It is the graph on the top left corner.

The results agree with what was expected. When the command "increase the speed" is sent to the Raspberry Pi, an increase in the power transmitted to the motor is recorded. And a power decrease is recorded for a command that decreases the speed.

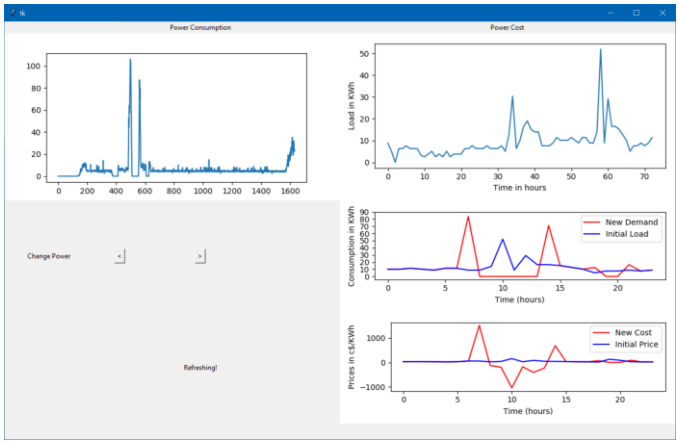

Figure 14 : Client interface when the command increases or decreases the speed of the motor.

B. Response to the automatic redistribution of the load algorithm.

This algorithm follows the new demand and commands the motor to spin at higher or lower speed depending on the new demand (in red in the graph of the load demand).

When the client presses the butt [4]on, he clearly says to the system that he wants to follow the advice of the power provider by adapting his energy consumption to a better distribution throughout the day.

The client interface sends the command to the Raspberry Pi and the motor will follow the curve of the new demand, as shown in the figure 15.

The power consumed by the motor is displayed on the top left corner graph. The power consumed by the motor and the new demand have the same shape.

When the day is over the client will pay less for his energy consumption since he followed the advice of the power provider by adapting the energy consumption of his electric load.

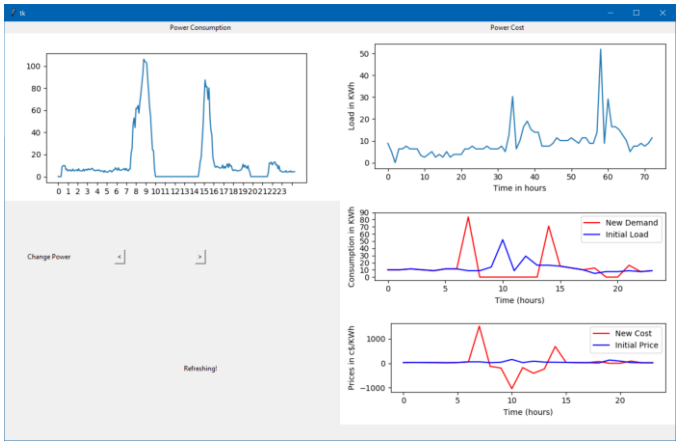

Figure 15 : Client interface when the electric motor follows the new demand curve.

IV. CONCLUSION

This project well implemented the automatic control of an electric load, to follow the new consumption demand obtained by a peak time rebate algorithm. All of this being controlled remotely via a client interface.

When the optimization algorithm is running, the power consumed by the electric motor follows the shape of the new demand, ensuring that the electric motor is adapting its speed depending on the new demand curve.

The client will receive a compensation for following the advice of the Power provider concerning his energy consumption.

In the future, being able to monitor and control loads, especially at a larger, consumer level, will lead to better understanding of how power is consumed and cost savings for customers.

V. REFERENCES

[1] S. Collier, «The Emerging Enernet: Convergence of the Smart Grid with the Internet of Things,» *IEE Rural Electric Power Conference*, pp. 65-68, 2015.

[2] L. D. Store, "SmartMeter Energy Consumption Data in London Households," 2019. [Online]. Available: https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households.

[3] S. S. a. E. M. S.H Pramono, «Internet based monitorig and protection on PV smart grid system,» *International Conference on Sustainable Information Engineering and Technology (SIET)*, pp. 448-453, 2017.

[4] S. B. T. Tuithung, «Remote Monitoring and Control of Smart Distribution Using Xbee Communication,» chez *2018 International Conference on Current Trends towards Convering Technologies (ICCTCT)*, Coimbatore, 2018.

[5] M. Y. a. H. Hejazi, «Design and Implementation of Things Based Smart Energy Metering,» *IEEE*, pp. 191-194, 2018.

[6] Zach, «howChoo - How to control a DC motor (or motors) using your Raspberry Pi,» [En ligne]. [Accès le 04 2019].

[7] «Wiki.t-o-f.info - Arduino > TB6612 1.2A DC/Stepper Motor Driver,» [En ligne]. Available: http://wiki.t-o-f.info/Arduino/TB6612. [Accès le 04 2019].

[8] ComEd, "Real-Time hourly pricing," 2019. [Online]. Available: https://hourlypricing.comed.com/live-prices/?date=20190408.

[9] C. A. E. Company, "Hourly pricing," 2019. [Online]. Available: https://hourlypricing.comed.com/live-prices/?date=20190408.