Energy Efficient Collaborative Sensing-based Design: Soft Keyboard Case Study

Mahsan Rofouei, Miodrag Potkonjak, and Majid Sarrafzadeh

University of California, Los Angeles

***Abstract*—With our approach for the synthesis and energy-efficient operation of a pressure sensor-based soft keyboard, energy is optimized at five levels of abstraction: (i) architecture of sensor systems; (ii) sensing schedule; (iii) sensor data processing; (iv) use of semantic information; (v) customization. We have been able to achieve up to a factor of 63 energy reduction in a keyboard case study over the standard. For example when each sensor senses several keyboard events and each event is sensed by three sensors, sensing and data processing results in a 12 to 42-fold energy savings depending on typing speeds. Customization techniques accounting for user typing speed and duration of typing pulses support another 1.5-fold savings. We demonstrate using a proof of concept prototype evaluation with five participants.**

***Index Terms*—E-Textiles, Sensing, Soft keyboard**

I. Introduction

The productivity of the user can be greatly improved by providing energy efficient, full scale and customizable keyboards that are light, low cost, and easy to transport and deploy. In addition minimizing the health risks associated with shared keyboards, E-Textile keyboards enable much better security and user authentication. Finally, the new keyboards can also collect a variety of statistics about the user that can be used for improving productivity and even possible detection of health problems. Our goal is twofold: (i) to design an E-Textile based portable keyboard and energy efficient algorithms for its use, and (ii) to study methods and techniques for design of layers in the sensing and processing stack of pressure sensor systems.

While one-to-one event-sensor mapping sensing seems to be a natural way to create soft keyboards, we show that a different design structure coupled with appropriate data collection and processing strategy is actually more energy efficient. Sampling the pressure for each keyboard key using a separate sensor is not energy efficient unless each key is equipped with equipment that can send an interrupt to the operating system. If a polling communication strategy is employed it is crucial to separate two events: one that indicates that any key is activated, and one that indicates that a particular key is pressed. By polling a single global sensor that indicates that any key is pressed, we can reduce the energy consumption. Once the event is detected, we query individual keys. Thus, we have developed a sensing architecture that groups individual sensors. The partitions are organized in such a way that by iteratively or simultaneously querying a small number of them, one can deduce which individual key is pressed.

The rate of sampling is dictated by two bounds. The first is a consequence of the need to conduct sampling at least at a rate at which two consecutive keys are pressed. We address this issue by conducting a study of representative subjects. The second bound is created by the need for additional sampling to identify a specific key using sensors that cover sets of geometrically connected areas below several keys. These two bounds are merged so that the allocated time between two consecutive samplings is sufficiently small to allow for the highest expected speed of typing. We next describe how we have addressed the sensing and processing stack of common embedded sensing systems.

Sensing system architecture. Our sensing architecture follows binary, many-to-many, and overlapping sensing strategies. We characterize each signal to deduce if it is or is not below an empirically measured threshold that indicates a particular event. We conduct additional measurements that are used for both increasing the robustness and identifying events (specific key pressing). Each key on our keyboard is exposed to two sensors in such a way that for each key we have a unique pair of sensors. The simplest potential realization of this type is a matrix of horizontal and vertical sensors.

Interleaved and coordinated data collection and processing. In many sensor networks applications data collection and processing are two separate phases. However, in the case of the soft keyboard, we merge these phases to enable flexibility and efficient search for event identification (diagnosis). We query a particular combination depending on the results of the already queried sensors. This strategy is possible only when data processing is faster than the maximal sampling rate. In our case we created this situation by restricting signals to their binary values.

Event detection and diagnosis. We separate event detection and diagnosis. This decision is mandated by highly non-uniform time between pressing two consecutive keys. Thus, event diagnosis is only performed when an event is detected.

Semantic information. We identified two types of relevant semantic information that can facilitate reduction of required energy. The first is any measure defined over the set of typed characters. Specifically, in our soft keyboard realization we use two simple measures: frequency and conditional frequency of characters. The measure enables us to better organize the search for the pertinent key by searching first for the keys that are more likely to be activated.

The second semantic measure is related to physiology and typing-level proficiency and has two components: 1) a minimum time between consecutive activation of two keys (of course, the keys may be identical) and 2) a conditional measure where we estimate and bind the minimal time that any key is activated after each particular key is pressed. This minimal time guides our sensor querying (sampling) schedule.

User customization. Because different individuals have significantly different speeds and characteristics of typing, our second semantic measure, minimal time between two consecutive activations, is customized for individual users and supports the creation of customized sampling schedules.

The rest of the paper is organized as follows: Section II presents related work in sensor networks, sampling strategies, typing, and keyboard design. Section III discusses E-Textiles and their characteristics. Sections IV and V describe architecture design and processing methodologies that we use for the E-Textile based keyboard. Experimental results are provided in Section VI. Conclusions appear in Section VII.

II. Related Work

Energy use is a critical system constraint in the design of sensor networks [17, 18]. [21, 22] have addressed sampling problems related to event-driven sensor sampling activation. [23] discusses distributed event-triggered sampling strategies and [24, 25] have developed techniques for dynamic sensor nodes so that required energy is minimized.

With respect to typing characteristics, keyboards, soft keyboards, and the use of semantic information for keyboard optimization, Fitt's model and extensions [3, 4, 5, 6] have been used to show human performance relationships related to movement such as a high correlation (often above 0.8) between the Fitt's index of moving difficulty and actual measured time to complete a task. Soft keyboard design [7, 8] and the use of semantic information for more effective text entry [9, 10] and keyboard optimization techniques [11, 12, 13, 14] continue to show human performance benefits.

E-Textiles have been introduced as a platform for pervasive computing [27]. [26] discusses a study on metrological properties of E-Textiles and provides pressure-resistance models for various E-Textile materials. Energy efficient routing in E-Textile applications [28] and designing efficient sensor structures and sampling techniques for E-Textile-based systems [29] are active research areas.

To the best of our knowledge this is the first effort to create a soft keyboard using smart textile and pressure sensors. It is also the first study that targets low energy soft-keyboards. Finally, the new approach is the first that uses global sensing for detection of localized events.

III. E-Textiles and Sensor Model

E-Textiles are composite yarns made of fibers coated with a conductive polymer. E-Textiles' foldable characteristics make them suitable for many portable and wearable applications. Next we describe how E-textiles can be used to create pressure sensors and how one can model their characteristics in the presence of pressure.

A. Sensor Fabrication

An E-Textile-based sensor has a three-stacked layer structure. The sensing material is sandwiched between two conductive pads. E-Textile acts like a pressure sensitive resistor. When force is applied, the resistance of E-Textile decreases. The conductive layer can be made of conductive fabrics, copper foil tape or conductive threads. To maintain the flexibility of the sensor and textile feel, we use conductive thread and conductive tape for the conductive layer.

B. Sensor Pressure-Resistance Characteristics

We model the force-resistance relationship as shown in Figure 1. Specifically, we show the curve for forces below 10N for a 2cm*2cm sensor. We use this curve to obtain resistance values corresponding to typing forces.

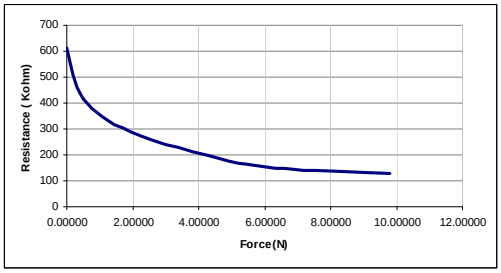

Figure 1. Resistance response as function of applied force.

C. Sensor Sensitivity with Distance

In fabricating a sensor array for mass production, the middle sensing layer is shared among all array sensors (See section IV.A. for details). One ramification is a dependency among readings of close sensors. When applying force on one sensor, a pressure is imposed on its surrounding sensors due to mechanical linkage. Therefore, we calculate the resulting pressure sensed on each location as both the direct and indirect forces. Hence our model includes the *neighboring effect*.

IV. Architecture Design and Processing Methodologies for E-Textile Based Systems

A. Sensing System Design: Solution Space Exploration

The main concern in designing sensing systems is to create a sensing architecture that captures all events and requires minimal energy. The common practice is to design a sensor array in which a single sensor is assigned to each sensing location with its dedicated signal line. However, energy savings may be accomplished through alternative sensing architectures.

In an *n* by *m* sensor array, where each sensor has its own signal line, *m*n* sensor readings are required to determine the location of asserted pressure. An alternative design mechanism is one where sensors on the same row/column share the same wire. In this architecture, a voltage is applied at the column of interest and the current at each row is measured. Using this reading mechanism the entire array is scanned column by column. This design scheme makes the routing much simpler since sensors on the same row/column share the

same wire, but still needs n*m readings.

We propose a multi-layer design where each layer consists of several sensing elements. The sensor array is constructed by placing the layers by considering the virtual locations of each sensor in other layers. Pressure asserted at the top layer is sensed in the layers below due to the nature of E-Textiles. Calibration can be performed to obtain equivalent pressure at all levels. This enables reconfigurable design based on application needs. Figure 2 shows this multi-layer design with *k* layers and the potential of maximum *m*n* sensors in each of the layers. However, based on specific design constraints and requirements, sensors in each layer can be combined to enable different sampling schemes and to form different size sensors. For example, they can be combined to produce coarse grain sensing at locations where detailed data are not needed (e.g. all sensors in one layer can be connected to form one sensor). Figure 2 shows several examples of such sensor combinations.

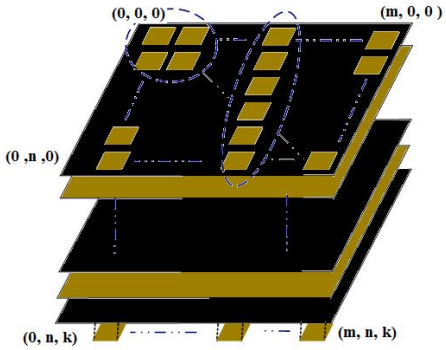

Figure 2. Multi-layer sensor array design with K layers and a maximum of m*n sensors on each layer

Using a higher level view of sensing elements in each layer, Figure 3 shows one way of constructing this sensing structure where each layer's sensing elements are combined into parallel sensing elements. Each sensing element here is the combination of multiple square sensors in one line. The placing of two layers is in such a way that the sensing elements on the layers become orthogonal to each other. The touching layer of these sensors are connected to Vcc and signals are read from each of the sensors on each layer. This design mechanism supports creating a sensing circuit with more sensing locations per sensor element and thus requires fewer readings. For *n* vertical and *m* horizontal sensing elements, only *n+m* readings are required for *n*m* sensors.

The spacing between sensing elements can be chosen based on desired effect of sensors on each other. Distance between sensors can be chosen so large so that the neighboring effect is negligible or chosen small enough so that only first neighbors of the sensor are affected. Sensing elements can be either drawn or sewn on to the virtual locations.

The inclusion of additional layers can either enable additional information processing techniques or provide features such as reliability for the sensing system. For example, two identical layers can be constructed in order to provide consistent results in the case of failure of sensors on one of the layers. However the design and decisions on combining sensors in each layer is non-trivial due to geometric constraints such as preserving the neighboring arrangement of sensors as discussed in Section V.B.

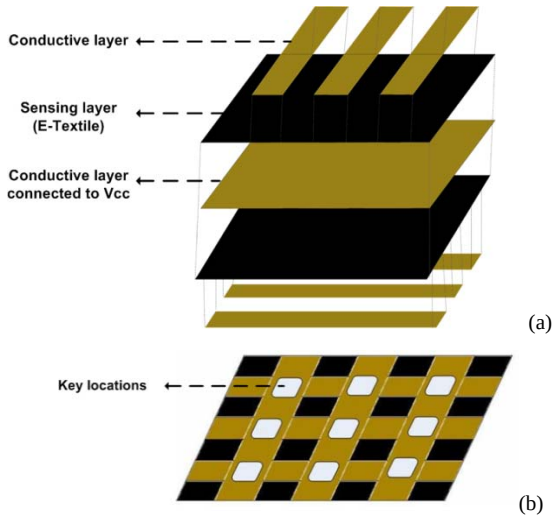

Figure 3. Two layered design (a) Layer arrangement (b) Virtual view of sensing array and sensing locations

1) Group Sensor Readings

Each sensor has a dedicated signal which is multiplexed and then read through A/D converters. However, in some situations it might be beneficial to combine sensors in groups and read through one A/D converter in order to save energy. One situation was described above. Other cases are when group readings can enable effective queries by allowing readings from a combination of sensors. To enable this capability, sensor readings can be combined as needed using electronic switches. For example, this can be accomplished using summing amplifiers as they keep the interaction between inputs at a minimum and produce an output voltage proportional to the algebraic sum of their inputs. The number and size of summing amplifiers in the sensing circuit is statically determined depending on application requirements.

B. Decoupled Event Detection and Identification

We decouple event detection from event identification. Traditionally, in order to find the location(s) of asserted pressure, readings of all sensor values at each sampling period is required. Consequences include high energy requirements and extremely fast sampling rates in order not to miss any events. With the use of group sensor readings we first create event detection sensors where readings from all sensors are combined. Hence, only one reading is required at each sampling period. Additional sensor readings are only performed in the case of an event. In this scheme the energy saving comes from reduced samples (use of A/D converters) while the energy consumed at a sensing sub-circuit remains the same.

Specifically, the design in example of Figure 3 would change to a three layer design where the third layer is a single sensor with the size of the whole sensing array and is in charge of detecting any event (key activation). Next we present techniques that reduce energy consumptions in this phase of signal processing.

C. Interleaved and Coordinated Sampling and Data Processing

We use a new paradigm of interleaved and coordinated sampling and data processing to obtain flexibility in managing sampling strategies for energy minimization. To avoid building a complex system of equations and to enable fast and robust reasoning, we use a binary sensing scheme. Our sensor architecture is designed to enables us to conduct processing using binary information about sensor readings due to an experimentally confirmed sensing model (described in Section V.D). Therefore, for event identification we use combinatorial search that rapidly eliminates possibilities that are inconsistent with the measurements from further consideration.

Only a subset of close sensors detect pressure changes. We perform a binary mapping from sensed values using a platform dependent threshold. Values above this threshold are mapped to 1 (0 otherwise). A group reading of n sensors after the binary mapping stage will result in a '1' if one of its *n* sensors is equal to 1. We take advantage of the above discussed group reading ability and binary representation of sensors to minimize the number of queries needed to identify events.

For each event detected, depending on the expected available time to perform processing, the number of sensor readings can be reduced. If the expected available time allows two epochs (units of time) for processing, using group readings, we can narrow down the search space by a factor of two, three, or more in the first epoch and then perform search in the reduced search space in the next epoch. This strategy can be extended to more epoch calculations using the same reasoning. Consider an example of two epoch processing of 8 sensors. In the first epoch, we perform a single group reading of 4 sensors. If the result is 1 we conclude that the activated sensor is within that group. Similarly if it is 0 it should be in the other group. Based on this result we perform search only on a group of 4. So in this example instead of performing 8 measurements we only perform 5. Different combinations and more epochs available can result in further savings.

The key question is how to determine the number of available epochs for measurement and reasoning. This number depends on the event detector sampling mechanism. A very fast sampling in the event detector sensor will capture events early enough to enable multiple epoch processing. Sampling at a very slow rate that is only fast enough to capture the presence of events, would allow one epoch processing. Consider the example in Figure 4 with an event happening in the event detection signal where point C is the end of the event. If the event detector sampling can assure the discovery of an event at an early point such as A, the time available for event identification is t_{AC}. However if the event detector sampling is only fast enough to guarantee to capture events at late points such as B, the remaining time for event identification becomes only t_{BC}.

To minimize the overall energy consumption of the system, we need to solve an optimization problem. More available epochs enable more optimized measurements and therefore less energy consumption in the event identification phase. On the other hand, increasing the number of epochs requires a higher sampling rate on the event detection sensor and therefore higher energy consumption in the event detection phase. This optimization problem can be solved depending on different parameters and constraints of applications. In the E-Textile keyboard examples these parameters consist of time of typing session and duration of a typing pulse.

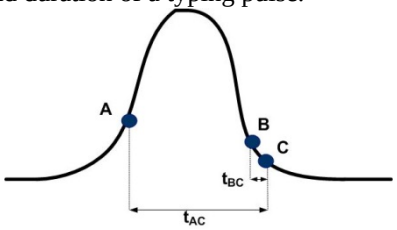

Figure 4. Sampling points A,B and C shown on the event detection signal

D. Exploiting Domain Specific Information

The use of semantic interpretations can enable optimizations both at the architecture and algorithm design levels. Domain knowledge provides information on rate of events, duration of events, location of events, and conditional probabilities of events. Rate of events dictates the sampling rate required for detection. These data, in conjunction with the duration of events, show how much processing time is available to perform interleaved sampling and data processing. Conditional properties of events provide information on how the sensor groupings can be performed. Location of events on the sensing array impacts the sensing circuit design. Studying user activity patterns can also provide information to optimize querying mechanisms. Finally, customization can be considered as semantic information that is related to a specific user and can also be used to optimize sampling strategies and minimize energy. Customization techniques used in the design of an E-Textile keyboard are described in Section V.E.

V. Portable E-Textile Keyboard Design

In this section we use the strategies presented in section IV to design and implement an E-Textile based foldable keyboard. We describe the prototype of the new keyboard. We also present the power consumption model that is the basis for our energy consumption measurements.

A. Power Consumption

1) Prototype Description

Figure 5 shows that our prototype is composed of a client part and a host part. In the client part, the pressure sensor array is scanned with the use of multiplexers. After the sensing data are acquired by the microcontroller, they are packaged and transferred to the host receiver side through a wireless RF circuit. The microcontroller used is MSP430f2274. The resolution of the A/D converter is 10 bits. The sampling rate is 10 Hz. The communication protocol used to transfer the data is SimpliciTI that uses CC2500 as the wireless communication chip. The SimpliciTI network protocol is a proprietary low-power radio frequency protocol targeting simple and small RF networks. This network protocol can be considered a complement to ZigBee and is more suitable for larger networks. The client is connected to the sensing circuit. Typing information is transferred wirelessly to the host side

(e.g. a smart phone, PDA or a laptop) where it is received by the RF chip.

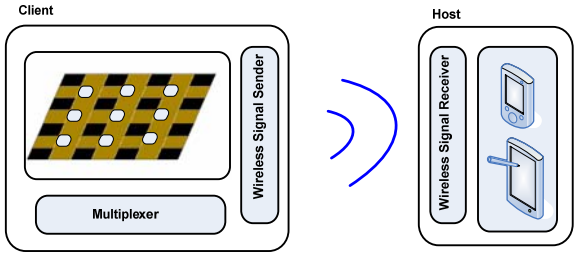

Figure 5. System prototype composed of client and host part

Since the applications of this system and similar systems would be in portable/wearable devices, we chose low power, lightweight components for our prototype design. We specifically selected the EZ430-RF2500 module from Texas Instruments for our processing and data transfer.

The prototype (Figure 6) is powered by two AAA batteries. A UART to USB converter (based on MP2010) is designed for interface compatibility.

1) Power Consumption Model

The total power consumption is composed of the power consumed in the sensing circuit and the transmission sub-system (1).

$$P_T = P_{Sens} + P_{com} \qquad (1)$$

The sensing power (P_{Sens}) consists of the power driven by the sensing circuit and sampling power. The former is calculated using resistance of asserted pressure. The latter is proportional to the power consumed by ADCs as each sensor reading is through a 10-bit ADC. As it can be seen in Table 1, P_{Sens} is dominated by ADC power. Hence group readings are beneficial as they only employ a single ADC unit. In the event of group readings the power consumed by summing amplifiers which is in the same order of sensing circuit are also added. Table 1 is calculated according to datasheets [15, 16].

Since data transmission power is similar to the ADC power we see that localized processing is beneficial and that minimizing the number of sensing readings dominates overall energy consumption.

TABLE 1. ENERGY CONSUMPTION OF SUB-MODULES.

SUB-MODULE	ENERGY(J)
ADC (PER SAMPLE)	9×E-9
DATA TRANSMISSION (PER SAMPLE)	7.2×E-9
SENSING CIRCUIT (AVERAGE PER SENSOR)	4.8×E-10

B. Sensing Architecture Design

Using a two-layer design (Figure 3) requires 17 columns and 6 rows to cover all keys. Rows and columns can be scanned simultaneously to identify the location of the pressed key. We simplify the querying mechanism to consider the extreme case of requiring one epoch (unit of time) for each reading. The number of queries is a linear function of the number of sensors in each group. Either columns or rows will dominate the querying time depending on the number of sensors. Our prototype needs 17 columns and 6 rows, so the querying is dominated by the columns. However, balancing the number of sensor in each group is more energy efficient. Note that 17 columns and 6 rows require 23 readings.

Consider a balanced architecture for sensing 102 keys (17*6) where 10 rows and 11 columns are used. Now, only 21 readings are required. However, due to geometrical constraints such as the need to preserve the neighboring arrangement of each sensor, a balanced design of 10 and 11 is not feasible. Figure 7 shows the sensor arrangements of 9 and 12 columns and rows that produce a semi-balanced architecture. Figure 7.a shows the arrangements of 12 rows instead of the original 6 rows. This layer would become the bottom layer of the keyboard. Note that this arrangement results in a unique pair of sensors for each key. Figure 7.b shows the arrangement of 9 columns. Imagining a virtual line in the middle of the layer, each two sensors from each side are merged to create one sensor. This imaginary line is aligned with the line breaking the 6 rows in 12 rows in the bottom layer. Therefore from Figures 7.a and 12.b we can see that we have 8 columns with 12 rows and 1 column with 6.

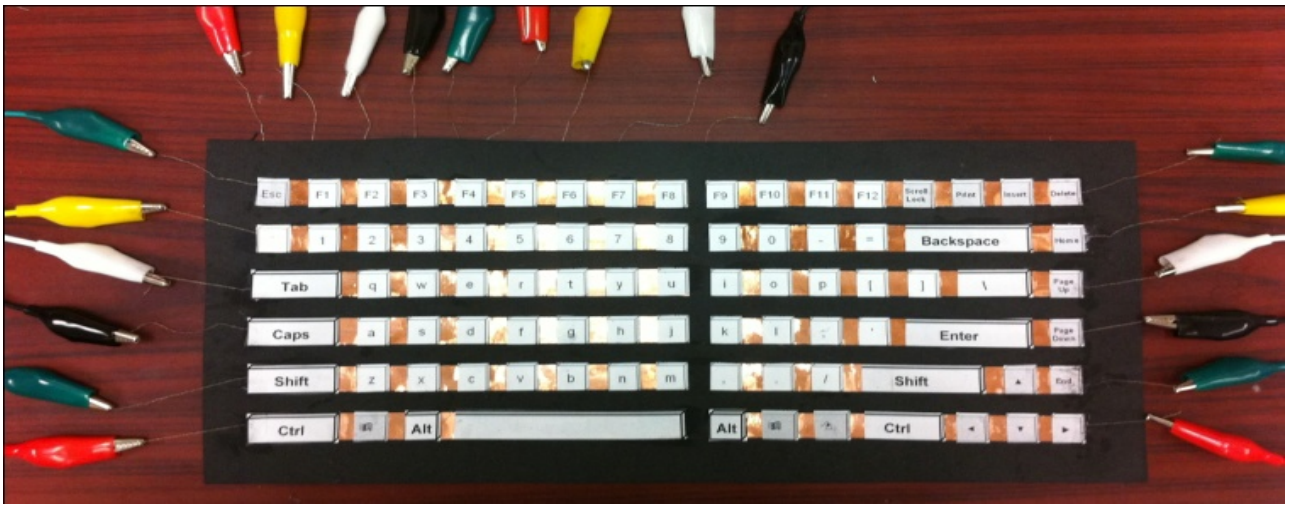

Figure 6. Prototype keyboard

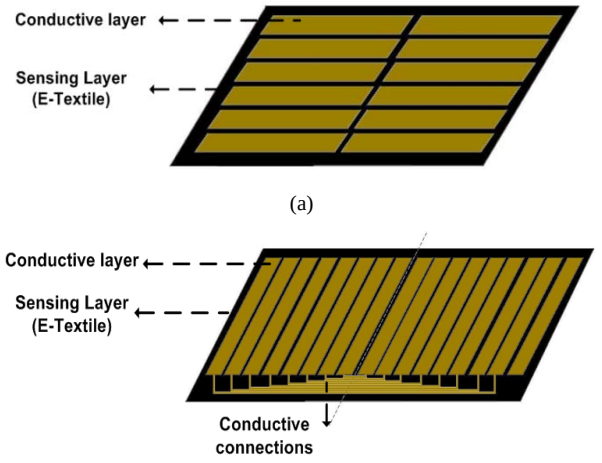

Figure 7. Balanced Sensor Array (a) Rows (b) Columns

The widths of the sensing elements are similar to those for available keyboards (around 1.3cm). The choice of the distance between parallel sensing elements impacts the detection algorithms. We choose the distance between consecutive sensing elements so that the neighboring effect is observable only on its direct neighboring sensors. In this design a key press would result in value changes on 6 readings (3 columns and 3 rows). By conducting experiments with five sensors placed within equal distances from each other and pressure asserted on the middle sensor, we found that placing the center of sensors within 1.5 cm spacing produces the desired neighboring effect. This distance was modified until the asserted pressure could only be sensed on the direct neighbors of the middle sensor. Note that our balanced design still needs only 21 (12+9) sensors readings.

C. *Decoupled Event Detection and Identification*

Using a summing amplifier we add all the signals from the columns to create an event detector signal. Figure 8 shows the event detector signal together with signals Column 1 and Columns 2 (represented by V1 and V2). The continuous sampling is performed only on this signal and further analysis on key identification is performed only in the case of an event. The main concern is to capture events early enough to allow sufficient time for event identification.

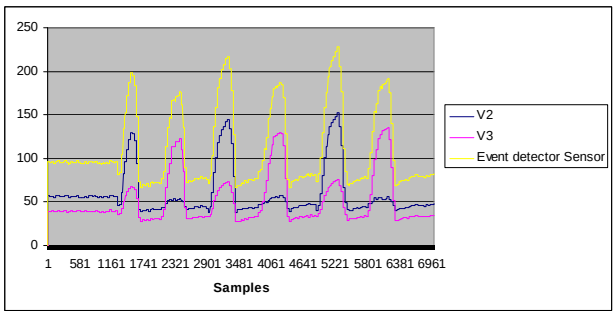

Figure 8. Event detector sensor, column 3 and 4 readings while pressing 'A' and 'S' keys consecutively. *X* and *Y* axis represent the number of samples and the pressure sensed.

Once there is no event detected by the event detector sensor for a specific amount of time (here 5 minutes), the keyboard is considered idle and thus the event detector sensor stops sampling to save energy. For activation of the keyboard after an idle period, the user uses an activation button on the keyboard.

D. *Interleaved and Coordinated Sampling and Data Processing*

Our prototype uses a sensing model where only the closest sensor and its two neighbors detect pressure changes. Figure 8 shows data from two adjacent columns on the keyboard (V_2 and V_3). Keys 'A' and 'S' (each on one column), are pressed consecutively. Due to the sensing model, in the case of activation on V_2, a relatively smaller activation is observed on V_3 and vice versa after applying the binary mapping stage, a single key press will result in either a pattern of three consecutive '1's or two consecutive '1's at the boundaries. The binary representation of 9 columns in the presence of a single key press at two different locations is:

000011100: Event on 6th column

110000000: Event on 1st Column

Using the interleaved sampling and data processing method, if the available time restricts the reasoning to be performed in one epoch, instead of performing *n* sensor readings to find a key press, reading every other sensor (plus 1, to cover both boundaries in the case of an even number of sensors) will suffice (*[n/2]+1*). For multiple epoch processing this can be done in the last step. For two epoch processing, after the search space is reduced by the order of two, three, or more in the first epoch, in the second epoch sensors can be measured in every other pattern. We assume that computations necessary for key identification to be performed in one epoch what is easily accomplished on 1 MIPS processor.

We show the efficiency of group readings and multiple epoch processing in improving the expected number of readings in the below example of four sensors (Figure 9).

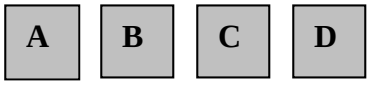

Figure 9. Four sensor example

In a 3 epoch available time, we arrange our readings as below:

Epoch 1:	Group reading of sensors A and B
Epoch 2:	Reading of sensor D
Epoch 3:	Reading of sensor C

In epoch 1 if the result of the group readings of sensors A and B is a '0', we can conclude considering the neighboring effect that the pressed key is D and we finish in the first epoch. If it results in '1' we proceed to epoch 2 and perform a reading on sensor D. Using measurement results of epoch 1, a '1' reading from sensor D means that sensor C is the pressed key since we sense its neighboring effect on its surrounding

sensors. Therefore in the case of C being our pressed key, readings terminate in epoch 2. Otherwise we proceed to epoch 3 where we perform a reading on sensor C. Using similar reasoning a '0' and '1' reading indicates sensors A and B as the pressed keys respectively. Therefore the expected number of readings reduces to 2.25 using the calculation below.

$$E(readings) = \frac{1}{4} \times 1 + \frac{1}{4} \times 2 + \frac{1}{2} \times 3 = 2.25$$

The traditional approach requires 4 readings. Using multiple epochs for reading but not performing group readings will result in the expected value of 2.5 readings.

Using similar reasoning we have calculated using a one epoch exhaustive algorithm, the optimal number of readings with group reading arrangements for different epochs for both rows and columns of our prototype keyboard. The expected number of readings required for rows and columns calculated using combinatorial search are shown in Table 2.

TABLE 2. COMBINATORIAL SEARCH FOR COLUMNS AND ROWS.

EPOCHS	NUMBER OF READINGS FOR COLUMNS (9)	NUMBER OF READINGS FOR ROWS (12)
1	5	7
2	3.55	4
3	3.22	3.66
4	3.22	3.66

The sampling mechanism in the event detector sensor determines the number of epochs available for processing. We set the longest interval without sampling in such a way that the time is minimal, but sufficient for both event detection and additional measurements required for key identification (equation (2)). If T_S represents time of typing session and T_P is considered minimum duration of a typing pulse, we can approximate the number of samples needed to be performed by the global event detector sensor as shown in equation (2).

$$N_{Samples} = \frac{T_S}{T_P - T_{Available}} \qquad (2)$$

Until now we assumed that a single key is pressed at a time. In some situations a user may press two or more keys simultaneously. We address the situation when two keys are pressed using an optimal algorithm, similar to the case when a single key is activated. Resolving the situation when three or more keys are simultaneously pressed induces high energy overhead. We leave this case up to the typist to correct such instances. Therefore, if three or more keys are pressed simultaneously no key presses will be inferred. Table 3 shows the requirements for our combinatorial search methods for 17 columns and 6 rows.

Comparison with Table 2 indicates that the new keyboard architecture is better in the first epoch. We actually have 8 columns of 12 and 1 column of 6 sensors. Using the calculations in Table 3 for 6 sensors we see that a 12 by 9 arrangement is better than 17 by 6 overall. Higher energy savings can be achieved if we change the column connection by splitting one column to create 7 columns of 12 sensors and 2 columns of 9.

TABLE 3. COMBINATORIAL SEARCH FOR UNBALANCED ARCHITECTURE

EPOCHS	NUMBER OF READINGS FOR COLUMNS (17)	NUMBER OF READINGS FOR ROWS (6)
1	9	4
2	4.70	2.66
3	4.23	2.66
4	4.17	2.66

E. Exploiting Domain Specific Information

We now show how the use of semantic information can be used to optimize sensing strategies and how we perform customization.

1) Semantic Information

We exploit the semantic information of the typing language in order to further reduce the system energy consumption. Results presented in Table 2 assume equal probabilities of events per sensor. Using semantic information columns and rows have unequal probabilities of being activated. We leverage this property to modify our querying scheme to find more likely events at an earlier time and thus reduce the total expected required number of readings. Consider Figure 9. Since different sensors have different probabilities of being pressed, it is beneficial to place the key with highest frequency in location D so that only one reading is required to identify it. Similarly, the next key with highest frequency is placed in location C and so on.

We used the relative frequency of letters in English from [1] to calculate probabilities for rows and columns of our prototype keyboard by adding probabilities of keys in each row/column. Using this frequency information we found the optimal sampling strategy for different epochs. Table 4 shows higher savings accomplished in the rows compared to columns. The reason is that, considering the standard layout of keyboards and our sensing architecture, keys are better distributed in the rows in terms of their frequency of occurrence relative to columns. One can also consider conditional frequency of letters to each other to calculate search strategies.

TABLE 4. COMBINATORIAL SEARCH WITH SEMANTIC INFORMATION FOR COLUMNS AND ROWS

EPOCHS	NUMBER OF READINGS FOR COLUMNS (9)	NUMBER OF READINGS FOR ROWS (12)
1	5	7
2	2.92	2.49
3	1.68	1.58
4	1.68	1.58

2) Customization

Users of keyboards have different typing-level proficiency and typing characteristics. This information can be used to perform customization on the sampling strategies. Note that only the event detection phase is impacted. We perform customization by exploiting typing speed information to adjust the event detection sampling. We also leverage customized information about latency of the next event as a conditional value that depends on the previous event.

The minimum time between consecutive activations of two characters is beneficial in adjusting the event identification sampling. For example, for a typist with the speed of 1 char per second there is no point in sampling every 0.1 second or less. From people used to standard keyboards, we collected data for the minimum time between two consecutive key presses using our prototype keyboard and found a range from 399 ms to 699 ms. We noticed relatively slower typing speeds on our prototype keyboard compared to standard keyboards which can be a minor limitation of such keyboards.

The conditional measure of the minimal time of an event after a known event guides our sensor querying (sampling) schedule. We measured the minimal time of an event after each of the alphabet characters. Figure 10 shows the minimum, median and maximum of this value for five participants who were asked to type a given text in a two minute session. Figure 10 indicates that while delays after characters are different across users, some characters have relatively longer delays.

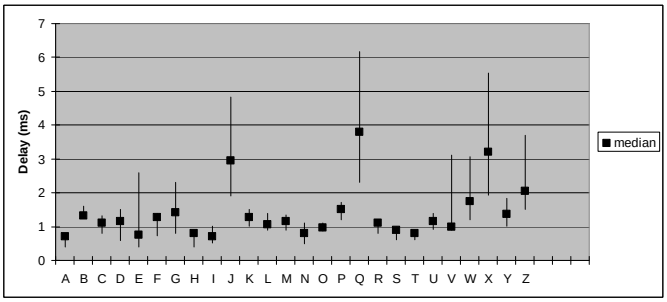

Figure 10. Conditional measure of minimal time of an event after a known event

VI. RESULTS

We gathered typing data from five participants and performed statistical analysis to find the necessary parameters for our proposed sampling scheme. One of these parameters is the duration of a key press pulse. Here we consider the minimum pulse duration computed over all people which is 250 ms. This parameter is used in determining the sampling frequency of the global sensor detector using equation (2).

We compare four different strategies (Table 5). Method 1 is energy results for non-intelligent sampling of all sensors at 10 Hz sampling intervals for key identification. This method is still better than a trivial non-optimized method since it uses a near optimized sensing architecture that requires only $n+m$ readings for $n*m$ keys. The savings from Method 1 stem from the methods in Section V.A. Method 2, uses statistical information on duration of typing forces to reduce sampling intervals on the optimized sensing architecture. In Method 3 we use our new combinatorial search in binary representation of signals with epoch time intervals. Methods 2 and 3 are based on concepts described in Sections V.B and V.C. Method 4 (based on using semantic information discussed in Section V.D) integrates semantic information into our combinatorial search mechanism together with statistical information about typing force durations on our near optimized sensing structure. The time of each session was 20 seconds.

To show the efficiency of our combinatorial search sensing, we show the different situations of Method 3 and 4 based on available epochs and their energy consumption. Tables 6 and 7 show that more epochs available, allow more optimized scheduling of measurements and therefore lower energy consumptions.

TABLE 5. ENERGY CONSUMPTION OF SAMPLING TECHNIQUES (NJ)

TYPING SPEED (CHARS PER SEC)	METHOD 1	METHOD 2	METHOD 3	METHOD 4
0.5	41.4	15.12	1.34	0.98
1.75	41.4	15.12	2.89	1.71
2.2	41.4	15.12	3.45	1.98
5	41.4	15.12	6.91	3.62

TABLE 6. ENERGY CONSUMPTIONS FOR COMBINATORIAL SEARCH IN BINARY REPRESENTATION (NJ)

TYPING SPEED (CHARS PER SEC)	1 EPOCH	2 EPOCHS	3 EPOCHS	4 EPOCHS
0.5	1.8	1.40	1.34	1.34
1.75	4.5	3.10	2.89	2.89
2.2	5.47	3.71	3.45	3.45
5	11.52	7.51	6.92	6.91

TABLE 7. ENERGY CONSUMPTIONS FOR COMBINATORIAL SEARCH USING SEMANTIC INFORMATION IN BINARY REPRESENTATION (NJ)

TYPING SPEED (CHARS PER SEC)	1 EPOCH	2 EPOCHS	3 EPOCHS	4 EPOCHS
0.5	1.8	1.19	0.99	0.98
1.75	4.5	2.41	1.72	1.71
2.2	5.47	2.85	1.99	1.98
5	11.52	5.58	3.63	3.62

As Table 5 shows, using our most optimized version that uses 4 epochs we are able to achieve savings from 12x to more than 42x in energy consumption depending on the typing speeds compared to the non- optimized version (Method 1). Compared to Method 2 which uses statistical data of typing force durations we achieve 4x to 15x savings in energy consumption. As expected, for faster typists the savings are less than for slow typists.

The lifetime of the system using Method 4 is more than 250 hours with two AAA batteries. This is significant energy reductions compared to current available foldable keyboards [19, 20] which offer 90 hours life time.

Our most optimized version is when we consider, customization in our sampling. We show two different customization techniques. First we use minimum delay between characters for each person. This customization has more benefit for slow typists (Customized A). In this method we consider both the duration of typing pulses and minimum delay between key presses. In the second customization we go further by using information from conditional measures of the minimal time based on characters identified (Customized B). The savings from customization is due to more efficient sampling in the event detection phase. Event identification energy remains the same. Table 8 shows the result of customization on three of the users and compares it with Method 4 which was the most optimized method without

customization.

TABLE 8. ENERGY CONSUMPTION OF CUSTOMIZATION SAMPLING TECHNIQUES (NJ)

USERS	TYPING SPEED (CHARS PER SECOND)	METHOD 4	CUSTOMIZED A	CUSTOMIZED B
1	0.59	1.66	1.24	1.11
2	0.39	2.16	1.89	1.62
3	0.69	1.54	1.08	1.00

Table 8 indicates that we can achieve up to a further 1.5x savings compared to our most optimized version before customization. Therefore considering altogether savings, we achieve up to 63x compared to the non-optimized method and 23x compared to Method 2. This means that the amount of energy expenditure for our most optimized version (Customized B) keyboard, is 63 times less than the amount of energy consumed in the non-optimized implementation of the keyboard.

With respect to ease of use, participants were not trained to use the E-Textile keyboard and they did not express any difficulty in using the E-Textile keyboard..

VII. CONCLUSION

We have developed an approach for synthesis and energy-efficient operation of a pressure sensor-based soft keyboard. We used E-Textiles as our sensing material due to their foldable and wearable characteristics and thus good targets for mobile and portable systems. We optimized energy at five levels of abstraction: architecture of sensor systems, sensing schedule, sensor data processing, use of semantic information, and customization. Our results indicated savings up to 63x in the overall system energy consumption compared to the system where each key is sensed by exactly one sensor. The procedure we used could likely be easily retargeted to other pressure sensor-based systems such as smart (medical) shoes, gloves, and bed sheets.

REFERENCES

[1] H. Beker, and F. Piper, Cipher Systems: The Protection of Communications. 1982.

[2] Edsger W. Dijkstra, The structure of the “the”-multiprogramming system, Proceedings of the first ACM symposium on Operating System Principles, pp.10.1-10.6, 1967.

[3] P.M. Fitts, The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, Vol. 47, pp. 381-391, 1954.

[4] I.S. MacKenzie, Fitts' law as a research and design tool in human-computer interaction, Human- Computer Interaction, Vol. 7, pp. 91-139, 1992.

[5] W. Soukoreff, I.S. MacKenzie, Theoretical upper and lower bounds on typing speeds using a stylus and keyboard, Behaviour & Information Technology. Vol. 14, pp. 370-379, 1995.

[6] S. Zhai, A. Sue, J. Accot, Movement model, hits distribution and learning in virtual keyboarding, SIGCHI conference on Human factors in computing systems, pp. 17-24, 2002.

[7] I.S. MacKenzie, S.X. Zhang, The design and evaluation of a high-performance soft keyboard, SIGCHI conference on Human factors in computing systems, pp. 25-31, 1999.

[8] M. Hunter, S. Zhai, B.A. Smith, Physics-based graphical keyboard design, CHI Human factors in computing systems, pp. 157-158, 2000.

[9] M. Silfverberg, I.S. MacKenzie, P. Korhonen, Predicting text entry speed on mobile phones, SIGCHI conference on Human factors in computing systems,pp. 9-16, 2000.

[10] J. Goodman, Venolia, K. Steury, C. Parker, Language modeling for soft keyboards, Intelligent user interfaces, pp. 194-195, 2002.

[11] S. Zhai, M. Hunter, B.A. Smith, The metropolis keyboard-an exploration of quantitative techniques for virtual keyboard design, ACM symposium on User interface software and technology, pp. 119-128, 2000.

[12] M. Raynal, N. Vigouroux, Genetic algorithm to generate optimized soft keyboard, CHI Conference on Human factors in computing systems, pp. 1729-1733, 2005.

[13] X. Bi, B.A. Smith, S. Zhai, Quasi-qwerty soft keyboard optimization, Conference on Human factors in computing systems, pp. 283-286, 2010.

[14] A. Gunawardana, T. Paek, C. Meek, Usability guided key-target resizing for soft keyboards, Conference on Intelligent user interfaces, pp. 111-118, 2010.

[15] http://focus.ti.com/docs/prod/folders/print/msp430-adc10.html

[16] http://focus.ti.com/docs/toolsw/folders/print/ez430-rf2500t.html

[17] D. Ganesan, B. Krishnamachari, et al. , Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks, Technical Report UCLA/CSD-TR 02-0013, 2002.

[18] J. Polastre , J. L. Hill , D. E. Culler ,Versatile low power media access for wireless sensor networks, Proceedings of the 2nd international conference on Embedded networked sensor systems, 2004.

[19] http://www.wirelessground.com/blfoblke.html

[20] http://www.onlypre.com/blfoke.html

[21] M. Malinowski ,M. Moskwa, M. Feldmeier, M. Laibowitz, J. Paradiso, CargoNet: a low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asychronous monitoring of exceptional events, Proceedings of the 5th international conference on Embedded networked sensor systems, pp. 145-159, 2007.

[22] S. Jevtic , M. Kotowsky , R. P. Dick, P. A. Dinda, C. Dowding, Lucid dreaming: reliable analog event detection for energy-constrained applications, Proceedings of IPSN, pp. 350-359, 2007.

[23] P. Wan and M.D. Lemmon, Event-triggered distributed optimization in sensor networks. In Information Processing in Sensor Networks (IPSN), pp. 49-60,2009.

[24] F. Koushanfar, N. Taft, M. Potkonjak, Sleeping coordination for comprehensive sensing using isotonic regression and domatic partitions, in: Proc. Infocom 2006.

[25] J. Liu, P. Cheung, F. Zhao, L. J. Guibas, A dual-space approach to tracking and sensor management in wireless sensor networks, Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, pp. 131-139, 2002.

[26] Z. Del Prete, L. Monteleone, R. Steindler, A novel pressure array sensor based on contact resistance variation: metrological properties. Rev. Scientific Instruments 72, 2001.

[27] D. Marculescu, R. Marculescu, N.H. Zamora P. Stanley-Marbell, P.K. Khosla, S. Park, S. Jayaraman, S. Jung, C. Luterbacj, W. Weber, T. Kirstein, D. Cottet, J. Grzyb, G. Troster, M. Jones, T. Martin, Z. Nakad, “Electronic textiles: a platform for pervasive computing,” Proc. IEEE, vol. 91, no. 12, pp.1993-1994, Dec 2003.

[28] J. Kao, and R. Marculescu. "Energy-aware routing for e-textile applications." In Proceedings of the conference on Design, Automation and Test in Europe-Volume 1, pp. 184-189. IEEE Computer Society, 2005.

[29] M. Rofouei, M. Potkonjak, and M. Sarrafzadeh. Energy e_cient e-textile based portable keyboard. In Low Power Electronics and Design (ISLPED) 2011 International Symposium on, pages 339{344. IEEE, 2011.