Supercapacitor Energy Buffering for Self-Sustainable, Continuous Sensing Systems

Meng Zhu Moeen Hassanalieragh Amal Fahad Zhuan Chen Tolga Soyata
Kai Shen

The University of Rochester
Computer Science Department
Rochester, NY 14627

Technical Report TR-995

March 2016

Abstract

Continuous, high-data-rate visual sensing and processing in the field is important for intelligent transportation, environmental monitoring, and area / context awareness. Rechargeable batteries in self-sustainable systems suffer from adverse environmental impact and fast aging. Advancements in supercapacitor energy density and low-power processors have reached an inflection point, where a data-intensive visual sensing system can rely solely on supercapacitors for energy buffering. This paper demonstrates the first working prototype of such a system, consisting of eight 3,000-Farad supercapacitors, a custom-designed 70-mW controller / harvester board, and a Nexus 7 tablet computer. We leverage the voltage-to-stored-energy relationship in capacitors to enable precise energy buffer modeling (no more than 3% error in time-to-depletion prediction). We find the supercapacitor self-discharge (or leakage) to be a minor issue in practice where the operating power significantly exceeds the leakage power. However, accurate energy budgeting must account for the variation of effective capacitance—particularly lower capacitance at lower voltages nearing energy depletion. Precise energy budgeting allows model-driven system adaptation (such as dynamic CPU configurations on low-power chipsets) and consequently realizes high, stable operational quality-of-service. Our working prototype has been successfully deployed at a campus building rooftop where it analyzes nearby traffic patterns continuously.

This work was supported in part by the National Science Foundation grants CNS-1217372, CNS-1239423, and CCF-1255729.

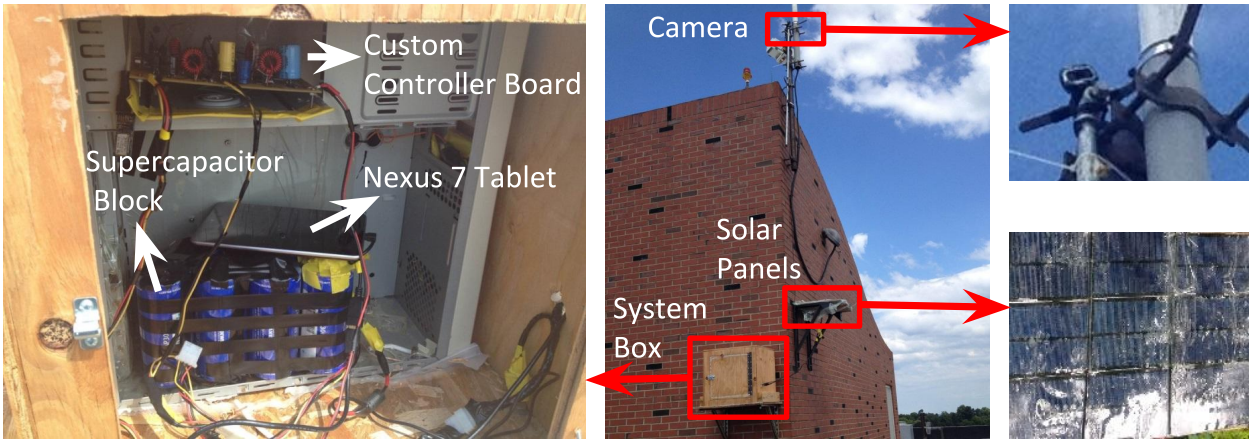

Figure 1: Picture of our deployed system on a seven-story building rooftop (center picture) that includes a camera, a block of solar panels, and a system box. The system box (left-side picture) contains a Nexus 7 tablet computer (without its internal rechargeable battery) sustained by eight Maxwell 3000 Farad supercapacitors (wrapped in black tapes), and a custom-built controller board (at the top of the box). The energy buffering capacity of the supercapacitor block is about 1.4 × that in the original Nexus 7 battery.

1 Introduction

Continuous sensing (particularly of high-data-rate visual information [2]) in the field enables various emerging applications. For example, continuous images are captured on or near a road [41] to analyze the traffic conditions, and consequently improve transportation safety and efficiency. In the domain of wildlife monitoring [29, 42], camera sensors continuously capture large volumes of data in the physical environment that is used to identify moving objects and activity patterns.

In these systems, data processing in the field, where data sources reside, is crucial for continuous intelligent sensing. First, by relying less on the unreliable (often wireless) communications in the field, the system could operate at high data processing capacity and with enhanced reliability. This is in contrast to capture-only systems that transfer large volume of raw data to remote processing sites which require high speed network connections. Second, instead of passively recording the scene, systems that are capable of processing the data locally could identify complex events and provide faster responses to emergent events (e.g., highway accident and congestion signaling).

Self-sustainability is critical for successful data sensing and processing in the field. Wire-free field systems are easy to install and maintain. A self-sustainable field node must live on ambient energy sources (e.g., solar cells) and energy buffering to power the system when ambient energy is unavailable (e.g., no solar power at night). Conventional rechargeable battery-based energy storage has drawbacks of negative environmental impact from their chemical compounds, and limited lifetime ($\approx 10^3$ charge cycles) that necessitates expensive maintenance in the field. As an energy buffering alternative, supercapacitors have practically-infinite lifetime ($\approx 10^6$ charge cycles) due to their energy storage mechanism [6, 46]. This implies far lower maintenance costs and environmental waste. Supercapacitors are also free of acids and other corrosive chemicals, furthering their environmental friendliness.

Supercapacitors have traditionally lagged far behind on energy storage density and therefore have only been used in ultra-low-power sensor motes [20,33,40,47] that run limited-function custom software and low-intensity operations. However, the past decade has witnessed major progress in

supercapacitor manufacturing technology, translating to substantial energy density improvements. Today, a commodity 3,000 Farad supercapacitor [5] at its rated voltage of 2.7 Volts stores about 11 KJoules of energy. Multiple 3000 Farad supercapacitors can store sufficient energy to sustain 12-hour continuous data-intensive operations at 1 Watt. In the mean time, mobile processors have advanced to levels that permit computation intensive operations with very low power consumption. These two simultaneous progresses have reached an inflection point today that it is now possible for a field system to use supercapacitors as its sole energy buffer and conduct high-data-rate sensing and analysis. The resulted elimination of rechargeable batteries would make field-deployed systems much more reliable, easier to maintain, and more environmentally friendly.

The stored energy in a capacitor can be theoretically modeled as $E = \frac{1}{2}CV^2$, where C and V are the capacitance and terminal voltage respectively. This model facilitates precise energy budgeting and enables a field-deployed system to operate intelligently at optimal *stable* quality-of-service while maintaining energy sustainability. In contrast, precise energy budgeting is much more challenging for rechargeable batteries [10, 45] due to their complex energy storage mechanisms and relatively fast aging.

This paper presents our design and construction of a software / hardware platform (pictured in Fig. 1) for continuous sensing on supercapacitor-sustained systems. Our hardware platform includes a custom-built 70 mW controller board that regulates between solar power input and energy buffering and consumption. By addressing the self-discharge (or leakage) and voltage-dependent effective capacitance issues in supercapacitors, we enable precise energy buffer modeling and time-to-depletion prediction. We further exploit adaptive features in low-power computer systems and continuous sensing applications to optimize the application quality-of-service under specific energy budgets. Our experimental study utilizes high-speed image capturing and processing applications [19, 48] on a real deployment of vehicle traffic analysis in our campus and realistic data traces of wildlife camera traps [42].

2 Related Work

Field sensor systems have been studied extensively for over a decade [12,18]. Energy self-sustainability is a critical issue for these systems. Two particular factors distinguish our work from these prior efforts: First, existing systems typically support only small amount of data capture using temperature, motion, light, and humidity sensors among others, but fall short of incorporating high data rate sensors (e.g., high-speed, high resolution camera). High-data-rate sensing and processing in the field is important for emerging applications such as intelligent transportation [41] and wildlife camera traps [29, 42]. Second, our use of supercapacitors as the sole energy buffering mechanism provides critical benefits of reliability, low maintenance, and environmental friendliness.

Energy modeling of rechargeable battery-based systems have been investigated in the past. Zhang et al. [45] calibrated a battery discharge curve to produce the mapping between observed voltage and battery energy capacity. However, they cautioned that the mapping requires per-battery calibration and may also lose accuracy due to relatively fast battery aging. Dong and Zhong [10] leveraged the availability of current sensors on some battery interfaces and expanded the modeling rate through software-level event-driven power models. In comparison, the state of a supercapacitor can be more easily modeled from its observed terminal voltage, which enables precise energy management and optimization at the system level.

Although supercapacitors have been widely used in industry (e.g., automotive [21, 43], elevators [35]), their uses in computer systems have been limited so far. Wang et al. [39] evaluated the

effectiveness of supercapacitors in data centers as secondary, short-duration energy buffers when the power load drops temporarily. Work by Jiang et al. [20], Dutta et al. (Trio) [11], Zhu et al. [47], and Renner et al. [33] utilized supercapacitors to support low-power sensor motes (typically 10s of milliWatts or less power). We also recognize that hybrid energy storage (containing both batteries and supercapacitors) offers stronger energy density and is particularly attractive for high-energy-use systems like electric vehicles [43]. Our work shows that supercapacitors alone are able to sustain off-the-shelf tablet computers and high-data-rate sensing applications for practical field operations. This paper addresses the hardware platform and software support to enable such systems.

The modeling and management of supercapacitors have been addressed in the past. Zhu et al. [47] and Renner et al. [33] proposed predictive energy management with supercapacitors for sensor motes. Weddell et al. [40] explained that supercapacitor self-discharge (leakage) behavior is dependent on its internal state. Hassanalieragh et al. [16, 17] described the circuit hardware design of solar power harvesting into a supercapacitor energy buffer. In the context of supporting continuous, high-data-rate sensing and processing in the field, our work studies the practical impact of supercapacitor leakage and effective capacitance issues, as well as the interplay between precise energy modeling and the energy-constrained adaptation of continuous data processing systems.

There is substantial prior work on managing low-power or energy-constrained systems. Flinn and Satyanarayanan [14] demonstrated profiling-based application adaptation to conserve mobile system energy. TinyOS [18] is an operating system construction that was tuned for very small system images on low-power sensor motes. Carroll and Heiser [8] as well as Pathak et al. [31] studied energy profiling and modeling in smartphone systems. Challen and Hempstead [9] argued that heterogeneous, low-power systems must adjust the component configuration in an agile fashion. Min et al. [27] devised a prediction framework that estimates the power impact of continuous sensing applications at installation time. The unique contribution of this paper is that we make the first demonstration that self-sustainable systems for continuous, high-data-rate sensing can rely solely on supercapacitor energy buffering.

3 Prototype System Construction

We design and construct a supercapacitor-sustained system platform for continuous sensing. Fig. 2 illustrates our system architecture. Key components of our system are a Nexus 7 tablet computer, a supercapacitor block for energy buffering, and a custom-built controller board to regulate power and supply critical power / energy statistics for software system management.

3.1 Application Workloads

A motivating application area for continuous data sensing and processing in the field is intelligent transportation [41]. These systems capture and analyze data streams of live road conditions for better traffic safety and efficiency [41]. In particular, a traffic trajectory analysis application [19] can monitor vehicle movements and extract their trajectories to enable the analysis of roadway conflicts [36], dangerous pedestrian violations [7], and the effectiveness of new traffic treatment [44]. We deployed the traffic trajectory application on a building rooftop-based field system to study the street and parking lot traffic condition in our campus. Fig. 3a shows an image snapshot captured during our deployment that shows annotated vehicle trajectories.

Another important application area is environmental camera traps [29]. Specifically, a field system may capture and monitor a live stream of images from a camera. We run the ZoneMinder

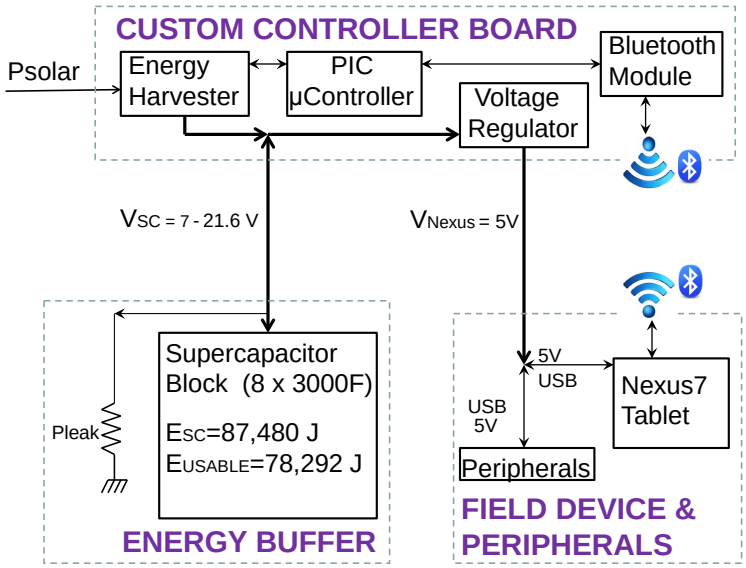

Figure 2: System architecture including the supercapacitor block for energy buffering, Nexus 7 for energy consumption, and a custom-built 70mW controller board that regulates between solar power input and energy buffering / consumption.

application [48] to difference each newly captured image against a background scene computed by averaging a series of recent past images. Such differencing can detect motion or new objects (like roaming animals in the park or floating logs in the river). We have acquired a set of field videos collected by ecologists to assess the predation risk for crayfish and analyze how the local habitat structure and landscape context might moderate such predation risk [42]. Fig. 3b provides an image snapshot that shows a raccoon searching for crayfish.

3.2 Computing Platform

High-speed processing of high-frame-rate images requires substantial computational capability that is not available on simple mote-based wireless sensor nodes. Data processing applications (e.g., OpenCV [30]) and associated tools also desire a conventional software environment (e.g., Linux). We selected the Nexus 7 tablet computer containing a Tegra3 quad-core mobile processor [37] in our prototype system. We disabled the display and other unnecessary components to minimize its power consumption. Tegra3 allows multiple CPU speed options from a single active core at 102 MHz to four active cores at 1.2 GHz with a wide dynamic power range. Total idle system power is 0.54 Watts using a single active core at 102 MHz. With four active cores at 1.2 GHz, 35 frames/sec operation of ZoneMinder, the system consumes 3.35 Watts and a CPU-intensive microbenchmark consumes 5.13 Watts.

We support field operations for the Nexus 7 on solar energy harvesting. Some applications only work during the day when there is sufficient light for camera operations. Others operate continuously through the night when collecting data over night is possible. We target the support of at least 14 hours of system operation on buffered energy (with no power harvesting). This can be a dark night or an overly cloudy day.

(a) Traffic analysis

(b) Environmental camera traps [42]

Figure 3: Image snapshots of our camera sensing workloads.

3.3 Supercapacitor Provisioning

We explain four critical issues in supercapacitor provisioning.

Energy Buffered in the Supercapacitor block (E_{SC}): A supercapacitor can store energy in the amount of $E = \frac{1}{2}\, C \times V^2$, where C is the capacitance of the supercapacitor, and V is the voltage at its terminals. Ignoring the manufacturing tolerances among supercapacitors, serially connecting multiple supercapacitors to create a supercapacitor block will force them to pass the same current and charge each individual supercapacitor up to the same voltage, and consequently store the same amount of energy in each individual supercapacitor. We choose a serially connected block of eight Maxwell BOOSTCAP BCAP3000E [5] supercapacitors in our system, with a rated voltage of 2.7 Volts, and a rated capacitance of 3,000 Farads for each. Therefore, this block of eight BCAP3000E's can store a total energy in the amount of

$$E_{SC} = 8 \times \frac{1}{2} \times 3000 \times (2.7)^2 = 87,480 \,\text{Joules} \tag{1}$$

This block occupies a volume of 8"×7"×7" and weighs roughly 11 lbs.

Serial vs. Parallel Multi-supercapacitor Layout: Multiple supercapacitors can be connected in different topologies [23]. In a serial setup, multiple supercapacitors are connected in a topology resembling the connection of multiple batteries in a flashlight, yielding a high voltage at the terminal. Our setup of eight serially-connected supercapacitors result in a block terminal voltage of $8 \times 2.7 = 21.6$V when each supercapacitor is fully charged. In a parallel setup where the terminals of all supercapacitors are connected together, the terminal voltage is 2.7 V at full charge. Since our Nexus 7 system requires 5 V to operate, serial vs. parallel topologies demand very different voltage (DC-DC) converters. Our preferred serial setup can operate at much lower currents, drawn from the regulator shown in Fig. 2. Alternatively, a parallel setup would require working with significantly lower voltages, necessitating much higher operating currents to deliver the same power output (i.e., $P = V \times I$). Higher current levels are undesirable since they cause higher energy losses on the copper wires connecting the circuit ($P_{loss} = I^2R$ denotes the power consumption on a copper wire with a resistance of R).

Maximum Supercapacitor Block Voltage (V_{SCmax}): Rated terminal voltage of 2.7 V should not be exceeded for each BCAP3000E supercapacitor for safe operation [6]. If any supercapacitor exceeds 2.7 V, this overcharged region would cause extreme aging in the supercapacitor [6] and overcharging above 2.85 V for more than a minute could result in safety concerns [5]. Therefore, the maximum allowed voltage for eight serially connected supercapacitors

is $V_{SCmax} = 2.7 \times 8 = 21.6\,\text{V}$, provided that each supercapacitor is individually voltage-limited (at 2.7 V) by a precision zener-diode.

Minimum Supercapacitor Block Voltage (V_{SCmin}): The Nexus 7 system requires a constant 5 V supply. We limit the minimum supercapacitor voltage to 7 V through our software control but allow a safety margin down to 3.5 V.

3.4 Custom Controller Board

We describe the design of our custom-built controller board design (depicted at the top of Fig. 2). The total power consumption of our custom controller is $\approx 70\,\text{mW}$ including Bluetooth and the regulator quiescent power.

Energy Harvester: We use a standard switching DC-DC converter design called SEPIC [38]. It is able to harvest solar energy in the voltage range of 7-20 V and store the energy into supercapacitor block in the voltage range of 3.5-21.6 V.

PIC μController: We use PIC16F1783 μController to control the duty cycle of the harvester based on the well-known maximum power point tracking (MPPT) technique [13] to extract the maximum amount of energy from the solar panels. It also sends the supercapacitor voltage and other power/energy related information to the Nexus7 tablet.

Bluetooth Module: Our custom controller board communicates with the Nexus 7 through the Bluetooth connection. The average power consumption of the Bluetooth unit is 10 mW.

Voltage Regulator: We use a SEPIC DC-DC converter controlled by LTC1624 switching controller [25] to provide a 5 V regulated voltage to the Nexus 7 tablet. The regulator is able to achieve high efficiency with low quiescent power consumption.

4 Supercapacitor Energy Modeling

The limited energy storage on the supercapacitors requires careful energy budgeting and utilization. Motivated by the simple voltage-to-energy relationship of $E(V) = \frac{1}{2}CV^2$, we enable precise energy buffer modeling and time-to-depletion prediction on supercapacitor-supported systems. Specifically, assuming an incoming power supply of P_{supply}, a portion of which is consumed by the system without being buffered by the supercapacitors (denoted as P_{load}), we can calculate the time it takes to charge or discharge a supercapacitor block—

$$t = \frac{E(V_{start}) - E(V_{target})}{P_{load} - P_{supply}} \tag{2}$$

where V_{start} and V_{target} are the terminal voltage values of the entire supercapacitor block at the beginning and the targeted times, respectively. We call this the *time prediction model*.

We can also use the model to identify the proper power load such that the supercapacitor block can reach the target voltage in a specific time t—

$$P_{load} = P_{supply} + \frac{E(V_{start}) - E(V_{target})}{t} \tag{3}$$

We call this the *power budget model*. The accuracy of these supercapacitor models is critically affected by the following two issues.

4.1 Leakage Modeling

A supercapacitor is made out of activated carbon particles immersed in an ionic solution (electrolyte). When the terminal voltage of the supercapacitor increases, due to the impurities in the carbon-electrolyte interface, undesired redox operations take place in the electrolyte, causing the charge to decrease. This phenomenon manifests itself as decreased voltage at the supercapacitor terminals even without external load (i.e., self-discharge, or leakage) [34]. The leakage affects the supercapacitor energy modeling because the energy is consumed by the system load as well as the leakage.

We measured the leakage of our supercapacitors at different voltage levels. The leakage can be computed by subtracting the power load from the loss rate of the stored supercapacitor energy. To accelerate this measurement process, we applied a small power load of about 0.1 W during the measurement. The experiment lasted for more than 10 days to cover the full range (22.8 V to 7 V) of operating and overcharged voltages.

Fig. 4 illustrates the results of our leakage measurements. The leakage is significant (0.69 W) at the overcharged level of 22.8 V. As explained in Sec. 3.3, overcharged supercapacitors are unsafe to operate and we do not use them for the remaining experiments in this paper. With the normal operating voltages, the leakage is about 0.07 W at the top 21.6 V and decreases rapidly to negligible levels as the voltage approaches 19 V and lower. Without the additional 0.1 W load we added to measure leakage at different levels, we have observed another identical block of eight BCAP3000 supercapacitors to reach $\approx$ 16 V after about a year. Therefore, we conclude that, leakage is only a minor concern at the high end of the rated voltage of the supercapacitors for a field system (e.g., 19 to 21.6 V for our 8-supercapacitor block).

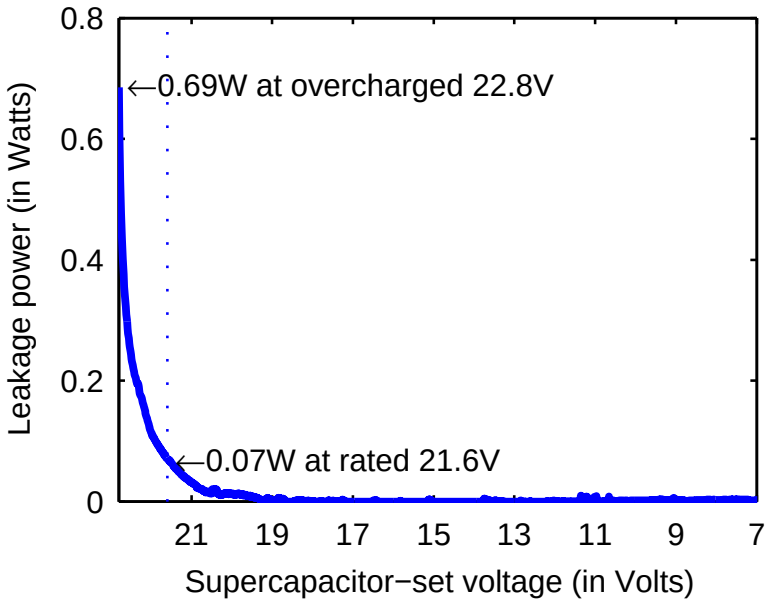

Figure 4: Self-discharge (leakage) power for our 8-supercapacitor set at operating and overcharged voltages.

Using the profiled leakage power at each voltage level, we augment our supercapacitor energy models to include leakage effects. For the time prediction model that predicts the time to reach a target voltage under a certain power load, we compute it through a discrete-time simulation. Specifically, we assume that, under a normal power load, the leakage power $P_{leakage}$ during a short

time duration Δt (e.g., 1 second) is constant and can be looked up through our profiled leakage power at each voltage level. Let V_t be the supercapacitor block voltage at time t, then the voltage at time $t + \Delta t$ can be computed by—

$$V_{t+\Delta t} = E^{-1}\Big(E(V_t) - \Delta t(P_{load} + P_{leakage} - P_{supply})\Big) \tag{4}$$

Our discrete-time simulation incrementally repeats this computation to predict the reachable voltage at an arbitrary future time, and can also predict the time to reach a target voltage.

Next we consider the leakage-aware power budget model that estimates the proper power load to reach the target voltage in a specific time. Built on the above time prediction model, we compute the power budget model using an approach similar to the Newton-Raphson method in numerical analysis. We start from two inaccurate bounding power load estimates—one too high (leading to shorter time to target voltage) and one too low (resulting in longer time to target voltage). We then take the middle of two bounding voltages and, depending on whether its time-to-target-voltage is too long or too short, decide on two new bounding voltages with a smaller gap. We repeat this process until the two bounding voltages are close enough to produce a low-error estimate.

4.2 Voltage-dependent Effective Capacitance

Our energy model is also affected by an important operational characteristic of supercapacitors: the energy storage capacity (i.e., the capacitance) increases at higher terminal voltages. This concept has been captured in prior work [28, 49] using a *three branch model*—energy is conceptualized to be stored in three branches (i.e., storage buckets). Leakage (introduced in the previous section) is the fourth—resistive-only—branch which is not associated with any storage, but, rather pure consumption. In the three branch model, the first branch consists of a fixed capacitor (C_0) and a voltage-dependent capacitor ($C_{0var} = a \times V$), where the term a signifies the degree of voltage dependence in terms of Farad/V. The first branch also has a very small Equivalent-Series Resistor (R_0). The other two branches are represented by two additional series resistor-capacitor pairs (namely, C_1, R_1, C_2 and R_2).

Our detailed analysis and measurements show that, for the power levels (up to 5 W) and the 8×3000 F supercapacitor block, lumping every fixed capacitor into a single fixed capacitor ($C_0 + C_1 + C_2$) causes a negligible error, while the effect of the *voltage-dependent capacitor* must be handled as a separate phenomenon. In other words, for a supercapacitor operating at a terminal voltage of V, the simplest approximation that yields reasonable accuracy is:

$$C_{eff} = C_0 + aV \tag{5}$$

While C_0 is the fixed storage capacity of the supercapacitor block, the parameter a is a direct implication of the three-branch supercapacitor model and cannot be ignored without risking noticeable modeling errors. We call C_{eff} the *effective capacitance* of the supercapacitor block. To demonstrate how noticeable this effect is, we have performed constant-current discharge experiments on our supercapacitor block. The effective capacitance at each short interval (where the capacitance is assumed to be a constant) is calculated by $C_{eff} = \frac{\Delta V}{\Delta t}/I$. Results in Fig. 5 show that, the nominal capacitance $C = 3000/8 = 375\,\mathrm{F}$[1] deviates substantially from the effective capacitance, especially at the critical, low-voltage range where the stored energy is near depletion. We also found that the effective capacitance for our supercapacitor block can be accurately modeled by $C = 310 + 3.8\,V$, where $C_0 = 310\,F$ and $a = 3.8\,F/V$ in Equation 5.

[1] due to our serial multi-supercapacitor layout as described in Sec. 3.3.

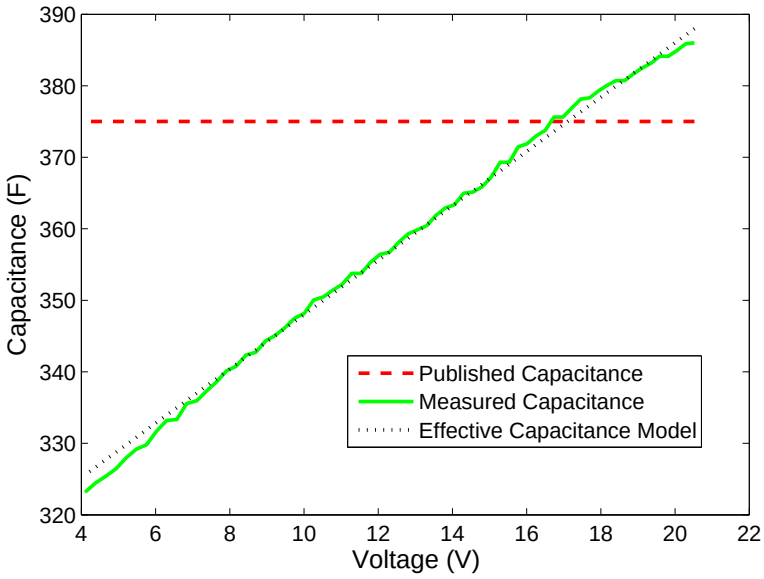

Figure 5: Effective capacitance at different terminal voltages. The capacitance is noticeably lower at low voltages.

The voltage-dependent effective capacitance requires changing the original supercapacitor energy model of $E(V) = \frac{1}{2}CV^2$ where the capacitance is assumed to be a constant. Specifically, we model the stored energy as the integral of power over time while the supercapacitor voltage decreases from V to zero:

$$E(V) = \int P\,\mathrm{d}t = \int v \cdot I\,\mathrm{d}t = \int_0^V v \cdot C\,\mathrm{d}v \tag{6}$$

$$= \int_0^V v \cdot (C_0 + av)\,\mathrm{d}v$$

$$\boxed{E(V) = \frac{1}{2}C_0V^2 + \frac{a}{3}V^3} \tag{7}$$

Using C_0 and a determined above, the total energy stored in our supercapacitor block can be computed from Equation 7 as $E(V) = 155\ V^2 + 1.2667\ V^3$.

5 System Energy Management

Our supercapacitor energy buffering mechanism provides limited energy storage with precise modeling and budgeting. Such a precise energy model enables efficient system management and adaptations under specific energy constraints. In particular, we can maximize the application quality-of-service while maintaining the energy sustainability of continuous data sensing and processing systems. Additionally, today's dynamic-power computing platforms offer opportunities for system and application adaptation under a given power budget.

5.1 Energy Sustainability and QoS Stability

Our system targets continuous sensing. Given a limited energy budget, we must plan and adapt its operation to avoid depleting the buffered energy prematurely. It is also desirable to maintain high, stable quality-of-service in the continuous data sensing and processing. In particular, it would be poor service if the system maintains a high-frame-rate operation over some periods of time but has to substantially degrade the frame rate at other time periods.

Our supercapacitor models presented in Sec. 4 enable us to apply feed-forward control [15] to determine an appropriate operational frame rate. Specifically, we set a target tuple of time (when the system can resume charging) and supercapacitor voltage (minimum voltage that can keep system running plus a small margin). Based on Equation 3, the operating power load (and consequently the operational sensing frame rate) can then be determined by the current supercapacitor voltage and the estimated solar input power.

Our system primarily relies on the model-driven feed-forward control but also includes feedback adjustments to prevent the buildup of modeling errors. We periodically monitor the supercapacitor voltage and adjust the system operating state accordingly. Note that both precise energy modeling (feed-forward control) and periodic adjustment (feedback control) are necessary to ensure high-quality, stable operations. Without precise energy modeling, periodic adjustment alone may still prevent the system from shutting down prematurely, but high energy modeling errors would require large QoS fluctuation for correction.

5.2 Model-Driven CPU Configuration Adaptation

At a given power budget, we need to identify the maximum quality-of-service (data sensing frame rate) in operational management. Modern CPUs offer a range of configurations with different CPU core counts and core frequencies. Multiple CPU configurations provide the opportunity for energy-constrained configuration adaptation. In particular, while a lower CPU configuration (fewer cores at a lower frequency) is generally more energy-efficient—consuming less power for the same rate of work, it suffers from a lower maximum data frame rate—possibly unable to fully utilize the power budget. An intelligent system must consider multiple possible CPU configurations to select the one producing the highest quality-of-service at a given power budget.

We build a power consumption model, characterizing the relationship between data frame rates and power consumption, under each candidate CPU configuration. Intuitively, the data rate is proportional to the resource utilization (CPU and memory activities) which is then linearly correlated with the system power consumption [4]—

$$P = P_{idle} + c_{capture} \times r_{capture} + c_{process} \times r_{process} \tag{8}$$

where P_{idle} is the idle power, $r_{capture}$ and $r_{process}$ are data rates (frames per second) for data capture and processing respectively. $c_{capture}$ and $c_{process}$ are coefficients in the linear model.

Work by McCullough et al. [26] pointed out that the linear power model may not be valid on modern multicore processors due to unpredictable resource contention and device intricacies. Varying resource contention, however, has not manifested in a significant way for our system. We attribute it to two reasons. First, continuous data sensing and processing applications exhibit stable resource usage patterns over time. Second, their data stream processing only reuses data with small distances (processing current or nearby frames) and therefore make more uses of the smaller, core-private caches than the larger, shared last-level cache. In a specific experiment, we

run our traffic trajectory analysis application (described earlier in Section 3.1) on the Tegra3 quad-core processor at 1.0 GHz CPU frequency with varying data rates from 1 fps to the peak of 19 fps. The processor performance counters report very stable Instructions-per-Cycle values (mean 0.510; standard deviation 0.013, or 2.6% of the mean) which suggests little variation of resource contention.

The Tegra3 chip in our Nexus 7 system allows a range of CPU configurations with different CPU core counts and core frequencies. We select four representative configurations in our study: I) 4 active CPU cores at 1.0 GHz, II) 4 active CPU cores at 640 MHz, III) 1 active CPU core at 475 MHz, and IV) 1 active CPU core at 102 MHz.

We have collected data on the power consumption under different data capture and processing rates and calibrated the power model for each of our four chosen CPU configurations. Fig. 6 shows our measurement and modeled power consumption for data capture+processing experiments.

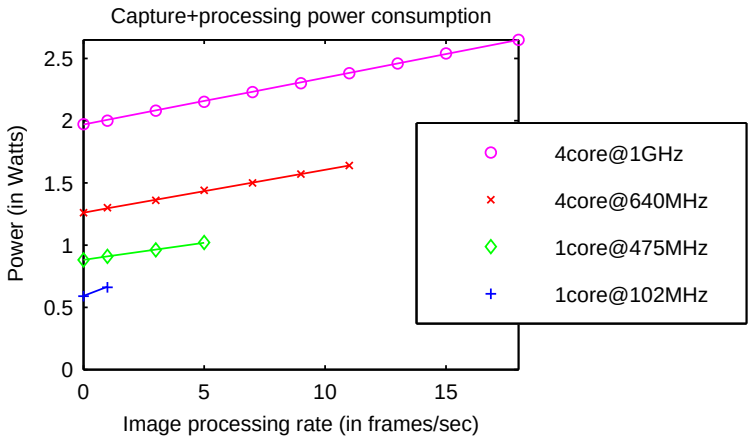

Figure 6: Data capture+processing power consumption at a range of CPU configurations and application frame rates. For each CPU configuration, the experiments end at the maximum achievable frame rate. We also show a linear fitting of the frame rate to power each CPU configuration.

To meet a power budget P_{load}, our model-driven CPU configuration adaptation works as follows. Under a specific CPU configuration, the power model in Equation 8 allows us to compute the application frame rate to meet the target power load—

$$r_{target} = \frac{P_{load} - P_{idle}}{c_{capture} + c_{processing}} \tag{9}$$

We perform such frame rate computation for each candidate CPU configuration and then choose the one supporting the highest data frame rate.

Delayed Burst Processing The CPU configuration adaptation presented above assumes a constant rate of data capture and processing. While a constant data capture rate is necessary for the continuous sensing purpose, the processing of captured data can be delayed. In particular, we explore *delayed burst processing* where the application alternates between two phases—a *capture-only phase* that stores the captured data without processing, and a *burst processing phase* that processes all previously captured data at the highest data rate allowed by the CPU configuration.

Under the delayed burst processing, only the burst processing phase needs a high CPU configuration while the capture-only phase can remain at an energy-efficient low CPU configuration. This

is more energy-efficient than the static CPU configuration selection that may have to remain at a high CPU configuration at all time. More fundamentally, such heterogeneous, two-phase computing exhibits much stronger power proportionality than a static configuration does [3, 9].

Under a pair of CPU configurations (a low configuration for the capture-only phase and a burst configuration for the burst processing phase), we compute the application frame rate r_{target} to meet the target power load P_{load}. Another important variable in this computation is the time ratio of the two phases through the execution. We use $(1-\delta)$:δ to represent the ratio between the capture-only time and burst processing time.

Let the power model parameters under the low and burst configurations be P^{low}_{idle} / $c^{low}_{capture}$ / $c^{low}_{processing}$ and
P^{burst}_{idle} / $c^{burst}_{capture}$ / $c^{burst}_{processing}$ respectively. Further let the burst processing rate be r_{burst}. We can then compute the target frame rate r_{target} and burst processing time ratio δ by solving the following two-variable quadratic equations—

$$\begin{cases} r_{target} = \delta \times r_{burst} \\ P_{load} = (1-\delta) \times (P^{low}_{idle} + c^{low}_{capture} \times r_{target}) + \\ \quad \delta \times (P^{burst}_{idle} + c^{burst}_{capture} \times r_{target} + c^{burst}_{processing} \times r_{burst}) \end{cases} \tag{10}$$

We perform such target frame rate computation for each candidate pair of low / burst CPU configurations and then choose the pair supporting the highest data frame rate.

Our delayed burst processing bears resemblance to computational sprinting [32] in terms of an elevated CPU configuration for bursty work. However, our different objectives (long-term energy efficiency for delayed burst processing vs. short-term fast responses for computational sprinting) demand different policy decisions. In particular, since the CPU overclocking hurts the long-term energy efficiency, burst processing at the highest CPU configuration is often the wrong choice for our approach. Instead, the optimal burst processing CPU configuration is typically the lowest configuration to fully utilize the power budget while the optimal capture-only CPU configuration is even lower to gain the maximum energy efficiency.

5.3 Model-Driven Duty-Cycle Management

CPU configuration adaptation is effective when the CPU power consumption dominates the overall system power. When this is not true (e.g., large static power of peripherals), we would need a more general approach to control the power / QoS tradeoff.

We take advantage of the suspension feature of low-power devices which typically consume minimum power during suspension (60 mW for our Nexus 7 tablet). Therefore the overall energy consumption of a device operation is approximately proportional to the amount of its active (non-suspension) time. This motivates a simple duty-cycle management that adjusts the active / suspension ratio to control the device power consumption. Let the power under active execution and system suspension be P_{active} and $P_{suspend}$ respectively. Let α be the active ratio. Then the average system power consumption is—

$$P_{load} = \alpha \times P_{active} + (1-\alpha) \times P_{suspend} \tag{11}$$

While being simple and effective, the drawback of duty-cycle management (compared to our CPU configuration adaptation approach in Sec. 5.2) is that the system would not be able to collect and process data during the suspension period.

5.4 Software Architecture and Implementation

In our software architecture, the operating system only provides the basic mechanisms for necessary information reporting and CPU configuration control (all exposed through the Linux `sysfs` interface). All energy modeling and system adaptation optimization work is preformed by the application at the user space. The application-controlled approach eases the integration of application semantics with energy modeling and optimization.

Specifically, we install Ubuntu 13.04 on the Nexus 7 and modify the Linux kernel to allow application privilege of directly controlling the CPU configuration. A target application is linked with a monitoring / control thread. It monitors the supercapacitor terminal voltage and determines the appropriate power load according to the energy model and desired operational condition (e.g., reaching a target voltage at a specific time). The application profile is then utilized to determine the appropriate CPU configuration or duty-cycle policy. Accordingly, the control thread exerts appropriate control—injecting delays into the worker threads or suspending/waking up the system.

6 Evaluation

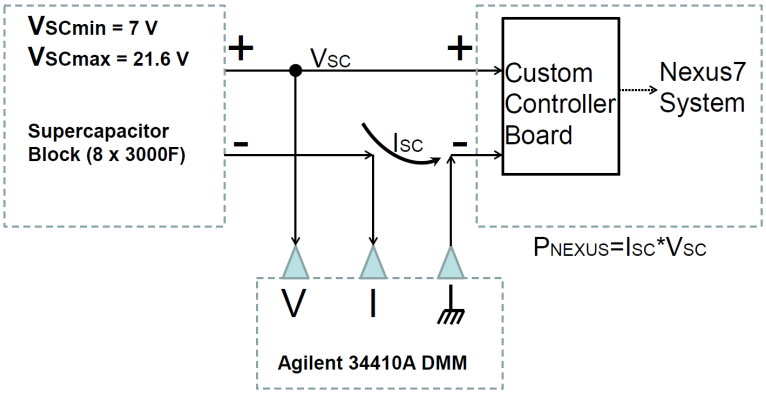

Figure 7: Agilent 34410A measurement setup to record simultaneous current and voltage values from the supercapacitor block to determine its power consumption (i.e., P_{SC}).

We built a measurement platform to acquire important metrics for our evaluation. Specifically, we connected an Agilent 34410A Digital Multimeter as shown in Fig. 7, to measure the supercapacitor voltage and current with high precision. The product of the voltage and current measurements yields our system power consumption. We logged the voltage and current read-outs to an external machine once every 10 seconds.

Our field device is capable of coarse-grained, internal logging of application processing performance (frames per second rate) and controller-reported voltage / current statistics. This coarse-grained logging, at the frequency of once every 10 minutes, consumes very little power. Therefore, for the field system's production run, we did not use the Agilent multimeter or the data logging machine to eliminate the need for additional power sources for these two devices.

Our evaluation targets two distinct goals of i) focusing on our core innovation of supercapacitor-sustained computing and ii) demonstrating the practical impact at the full-system scale. We take an incremental evaluation approach: Sec. 6.1 starts with an evaluation of supercapacitor energy modeling. Sec. 6.2 evaluates CPU configuration adaptation in a laboratory setup using only core

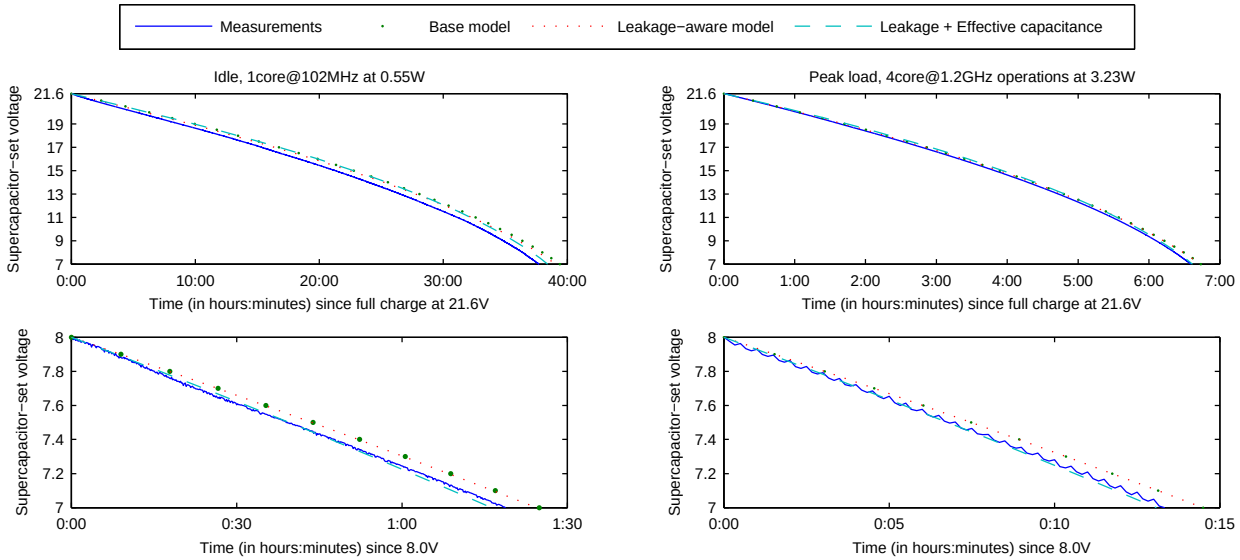

Figure 8: *Time prediction evaluation*—Accuracy of different supercapacitor energy models in predicting the time to reach a certain target voltage under specific power loads. The left two plots illustrate the results under idle load while the right two plots show the results under a high-load condition for the ZoneMinder-based environmental camera traps. The top two plots illustrate the results of modeling the operational time from the full charge of 21.6 V for the supercapacitors while the bottom two plots show the results of modeling from 8 V to 7 V.

system components (no solar panels or peripheral camera, using video as the input data source) under night-time conditions (eliminating the need for solar power modeling). Finally, Sec. 6.3 presents results from a real system deployment with a full set of peripherals in a 24-hour operational period. Applications and workloads were explained earlier in Sec. 3.1.

6.1 Supercapacitor Energy Modeling

We evaluate the accuracy of our supercapacitor energy models using the two energy model utilization scenarios described at the beginning of Sec. 4. First, the *time prediction model* predicts the time to reach a specific target voltage (e.g., effective depletion of usable energy) under a given power load. Second, the *power budget model* estimates the power budget that would utilize the stored energy in a certain amount of time. The power budget is then used to determine a proper application quality-of-service (QoS) in operations.

Evaluation of Time Prediction Model We compare the accuracy of three supercapacitor energy models—

1. the *base* model in Equation 2 without considering supercapacitor leakage or voltage-dependent capacitance,
2. the *leakage-aware* supercapacitor energy model presented in Sec. 4.1,
3. and the additional management of voltage-dependent *effective capacitance* presented in Sec. 4.2.

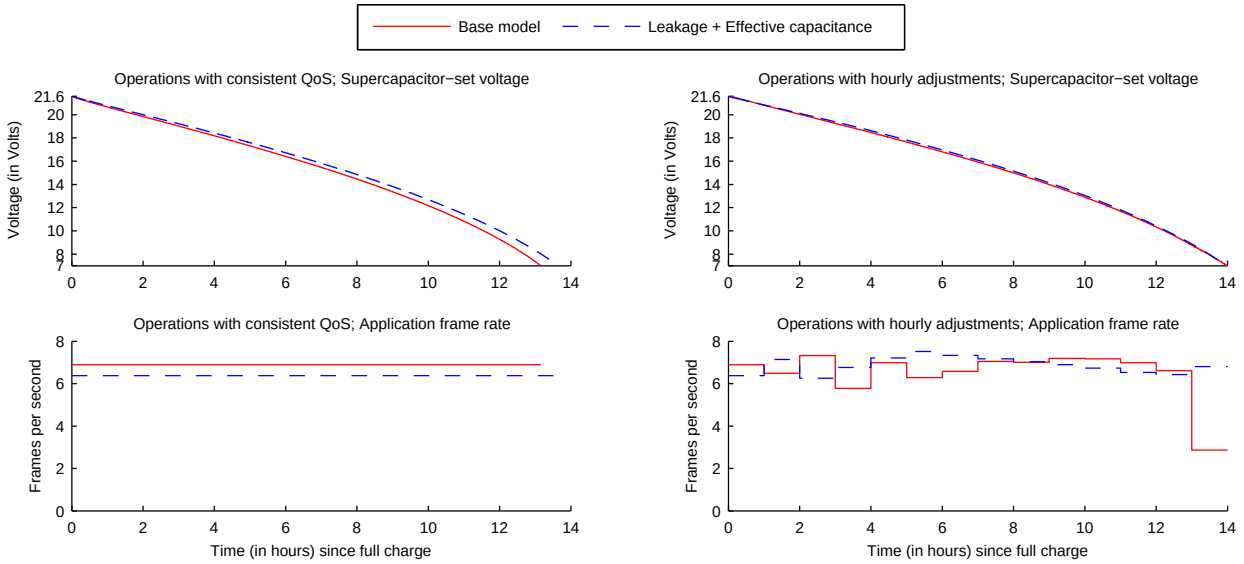

Figure 9: *Power budget evaluation*—Effectiveness of supercapacitor energy models in guiding application operational frame rates to utilize the stored energy by a certain amount of time. The experiments were performed for the traffic trajectory analysis application. The left two plots illustrate the supercapacitor voltage and application frame rates under operations of consistent frame rates. The right two plots show the results when the system makes hourly assessment of remaining supercapacitor energy and adjustment of application frame rates.

Fig. 8 shows the accuracy of the three models in predicting the time to reach certain voltages under two load levels (left / right columns)—0.55 W Nexus 7 idle load and 3.23 W peak operational load for the ZoneMinder-based environmental camera traps. We also show results in two modeling durations (top / bottom rows)—from the full charge of 21.6 V and from 8 V. The shorter-duration modeling accuracy is particularly valuable for systems that make periodic model adjustments in production.

We find that the base and leakage-aware models produce almost identical results under all evaluation cases. This validates our leakage profiling result in Sec. 4.1 that the supercapacitor leakage is insignificant during its normal operating voltages (in fact almost completely absent at 19 V and lower voltages). It produces no observable effect for practical system energy management purposes.

On the other hand, there is a clear effect of modeling the voltage-dependent effective capacitance, particularly at low voltage levels when the system energy is near depletion. Specifically for modeling the idle load from 8 V to 7 V, the effective capacitance model exhibits 3.3% error in predicting the time to energy depletion, compared to 7.2% error under the leakage-aware model without considering the voltage-dependent effective capacitance. For the peak load operation, the effective capacitance model reduces the modeling error from 8.8% to 1.9%.

The interesting curvy voltage patterns in Fig. 8 (bottom right) are due to the *charge redistribution* phenomenon at the supercapacitor block [49] which are observable when the power consumption levels exhibit some periodic fluctuation.

Evaluation of Power Budget Model We assess the effectiveness of supercapacitor energy models in guiding application frame rates to utilize the stored energy by a certain amount of time.

In this evaluation, we run the traffic trajectory analysis application to assure operation continuation during the night time while achieving the best application QoS. Our particular objective is that, starting a 14-hour dim period (part of the day without solar supply) with a fully-charged supercapacitor unit, we arrive at a target voltage by the end. We choose 7 V as our target voltage since, at 7 V, the supercapacitor can continue to support the system for another hour of processing after the dim period in case there is no solar input at all.

We compare two models: the base model and the effective capacitance model. For the base model, the above constraint yields an energy budget of 78,321 J which corresponds to an average power consumption of 1.554 W. For the effective capacitance model, the energy budget is 76,809 J and the average power consumption is 1.524 W. According to the static model driven CPU configuration described in Sec. 5.2, the configuration with 4 active CPU cores at 640 MHz yields the highest application frames rate, which is 6.9 fps for the base model and 6.4 fps for the effective capacitance model.

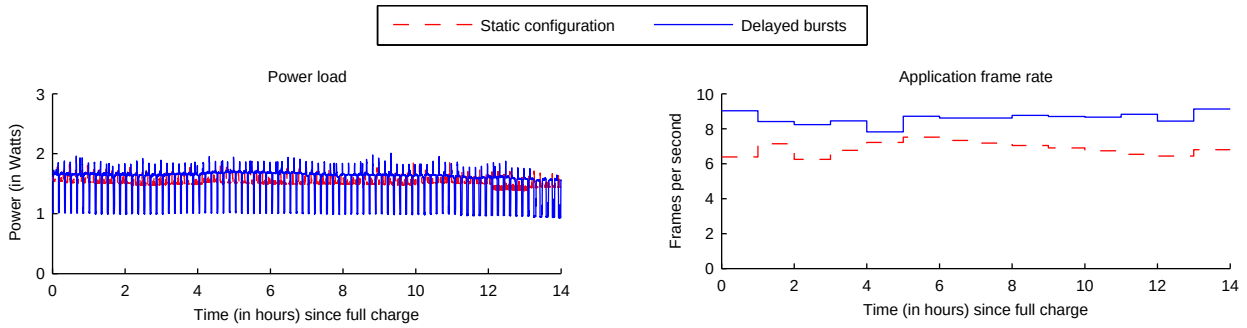

Figure 10: Power consumption and application frame rate under model-driven static CPU configuration and delayed bursts.

We consider two application operating approaches. The first approach uses the supercapacitor energy budget to determine the application frame rate at the beginning of the 14-hour dim period and maintains the frame rate consistently until the end. The second approach makes periodic (hourly in our experiments) assessment of supercapacitor voltage and buffered energy, and readjust the application frame rate accordingly. Such periodic model adjustments prevent error buildup and ensure on-time energy depletion, but they run the risk of producing unstable quality-of-service.

Fig. 9 shows that the precise energy budgeting yields benefits in both application operating approaches. For operations with consistent QoS, the base supercapacitor model exhausts the energy 51 minutes earlier than planned while the effective capacitance model reduces the error to 22 minutes. For operations with hourly adjustments, both models can support continuous operation for 14 hours as expected. On the other hand, the base supercapacitor model delivers unstable quality-of-service and its application frame rate plunges to 2.9 fps in the last hour. We attribute this to two reasons: 1) the voltage-dependent capacitance deviation is particularly pronounced at low voltages, as explained in Sec. 4.2, and 2) the cumulative effects of erroneous energy budgeting and control have to be reversed in a short amount of time. In contrast, the application frame rate under the effective capacitance model is consistently above 6.2 fps throughout the entire operation.

6.2 CPU Configuration Adaptation

A precise energy budget, together with a parametrized power consumption model, allow intelligent selection of the CPU configuration and dynamic application adaptation to achieve higher quality-

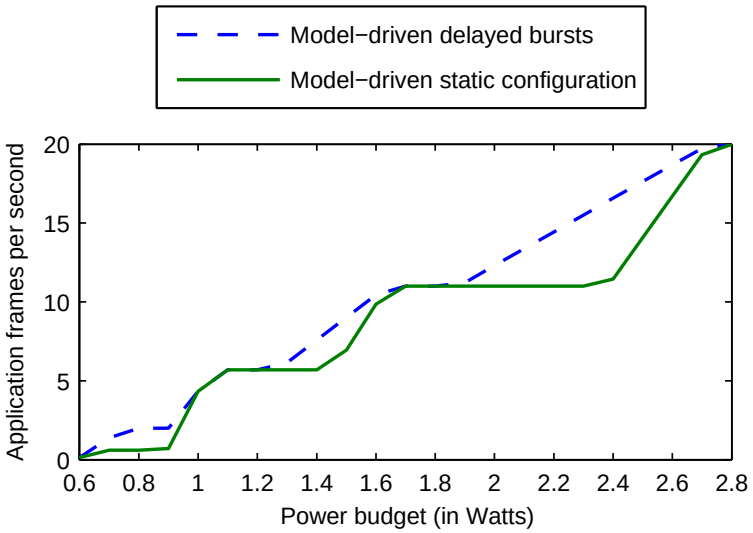

Figure 11: Quality-of-service under the two model-driven system adaptation approaches over a range of power budgets.

of-service. In our experimental setting, the system tries to sustain continuous operations during a 14-hour dim period while providing the best sustainable QoS.

We compared two different system / application management approaches presented in Sec. 5.2:

- The first approach used the model-driven CPU configuration to select the best static configuration that optimized for the QoS while ensuring continuous operation. Based on Equation 9, with a 1.52 W target power, the configuration with 4 active CPU cores at 640 MHz yielded the highest application frame rate, which was 6.4 fps.
- The second approach employed the delayed burst processing to further enhance the QoS. Using Equation 10 to compute the CPU configurations for the two phases of the delayed burst model (capture-only and burst processing), we derived 1 active CPU core at 475 MHz and 4 active CPU cores at 640 MHz respectively. We also computed that the ratio of (capture only)-to-burst time should be 0.15:0.85 ($\delta = 0.85$) and the application's achieved QoS was 9.3 fps.

While both approaches can ensure continuous operations through the 14-hour dark period, they exhibit different power consumption and application frame rate patterns. Fig. 10 illustrates these results. The pulse-shaped power consumption pattern in delayed bursts is due to the periodic switches between low-power capture-only and high-power burst-processing phases. The spikes are due to burst Bluetooth communication between the Nexus 7 tablet and the PIC controller.

In terms of application QoS, the delayed burst optimization (achieving 7.8 fps and higher) produced 30% enhancement over the static configuration (achieving 6.2 fps and higher). Fig. 11 compares the achieved quality of service (frames per second rate) between the two model-driven approaches over a range of power budgets. The delayed burst approach shows substantial benefits over the static configuration in some cases. The improvement on the frames/sec rate is at least 25% for power targets of 0.7–0.9 W, 1.4–1.5 W, and 2.2–2.5 W.

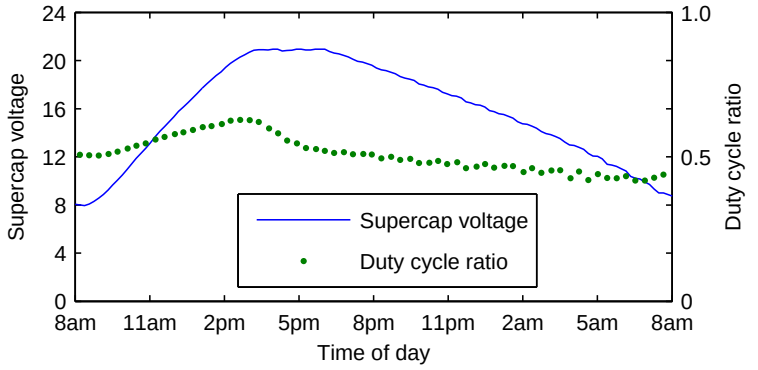

Figure 12: System duty-cycle ratio and supercapacitor voltage changes in a typical 24-hour duration of our real system deployment. The solar power supply lasts from 8 AM to 6 PM.

6.3 Real System Deployment

We deployed our system on the rooftop of a seven-story building (Fig. 1) at our university campus. Our deployed system monitors a street and parking lot in front of the building and analyzes traffic patterns continuously. The system has waterproof capabilities.

The goal of the system energy management is to: 1) stay above 8 V by 8 AM each morning when the energy harvesting resumes; and 2) provide the best stable QoS throughout the rest of the day. Accordingly, our power budget model calculates an appropriate power load based on the current supercapacitor voltage and the predicted solar power supply.

Compared to our laboratory experiments in the previous subsections, our real system deployment faces two additional challenges: First, we need a camera to capture live data. Unfortunately, commodity cameras (including the one we use) do not exhibit strong power-proportionality features that CPUs possess [24]. Their significant static power (about 1 W in our camera) renders CPU configuration adaptation ineffective in adjusting power. Therefore we use duty-cycle based management (Sec. 5.3) in our deployment. During the active (non-suspension) period, we set the CPU configuration to 4 core/640 MHz and fix the application processing speed at 7 frames/second. The system then adjusts its active duration over 5 minute intervals to achieve the average target power load.

Second, our deployed system supports 24-hour operations and it uses solar panels to harvest power during the day. Therefore our power management requires a model of the solar power supply. The solar power supply prediction has been investigated in the literature [1, 22]. While a precise night time energy model is crucial for system sustainability, daytime energy management does not have to be accurate (i.e., energy is not about to be drained). Our system predicts the solar power supply based on a simplified algorithm from [22]. Specifically, solar power in different time slots of the day is estimated based on a weighted average of observed solar power levels of the same slots in previous days.

Fig. 12 shows the result for a typical day of our rooftop deployment. The supercapacitor block voltage starts at 8 V in the morning and gradually increases until the block is full. As the buffered energy increases, so does the system duty-cycle ratio. After about 6 PM, the energy harvesting stops and the system starts to operate solely on its buffered energy. It maintains a stable duty-cycle ratio at night and reaches the target voltage (8 V) at the projected time (8 AM). We achieve stable night-time QoS by precise supercapacitor energy budgeting and power management. The

day-time QoS is less stable due to imperfect solar supply model. Fortunately, the QoS stability is less important during the day when the energy is abundant and the QoS is generally high.

7 Conclusion

This paper demonstrates the feasibility of using supercapacitors as the sole energy buffer in self-sustainable, high-data-rate field sensing systems. We constructed a prototype system with a block of eight 3000 F supercapacitors. This energy buffering mechanism, with typical solar power input, can continuously support a Nexus 7-based tablet computer running high-speed, high-resolution camera-driven data sensing and processing.

With a limited energy supply, we presented the software support to precisely model the supercapacitor buffered energy (within 3% error) and adapt the application for continuous operations and high quality-of-service. In particular, in dynamic-power, variable-configuration systems such as Tegra3 in the Nexus 7, we determined that, while the capture operation has to continue at a steady rate, the processing of the frames can be delayed and computed in burst mode, and therefore maximize the time for energy-efficient, low-power capture-only executions. This approach, we call *delayed bursts*, enhanced the application frames-per-second rate substantially compared to the optimal static CPU configuration (by over 25% at a wide range of power budgets).

Although leakage (self-discharge) is generally a concern for supercapacitor-based systems, our measurements show that this is a minor problem (0.07 W leakage at 21.6 V and quickly approaching zero at lower voltages). Such leakage power is not significant for high-data-rate sensing and processing systems. Our result is in direct contrast with low-power sensor systems [47] that report the impact of supercapacitor leakage to be much larger.

On the other hand, the simple supercapacitor energy model of $E(V) = \frac{1}{2}CV^2$ incorrectly assumes that the capacitance is a constant. In practice, the capacitance decreases gradually as the supercapacitor voltage decreases, necessitating a corrected energy model of $E(V) = \frac{1}{2}C_0V^2 + \frac{a}{3}V^3$. The corrected model implies an energy over-estimation using the simple model as the stored energy is near depletion (i.e., at low voltages). We addressed the voltage-dependent effective capacitance using a discrete-time-simulation-based supercapacitor energy model. With this corrected model, we observed a reduction in the time-to-depletion prediction error from 7–9% to 2–3%.

References

[1] M. I. Ali, B. M. Al-Hashimi, J. Recas, and D. Atienza. Evaluation and design exploration of solar harvested-energy prediction algorithm. In *Conf. on Design, Automation and Test in Europe (DATE)*, pages 142–147, Dresden, Germany, Mar. 2010.

[2] P. Bahl, M. Philipose, and L. Zhong. Vision: Cloud-powered sight for all. In *the Third Intl. Workshop on Mobile Cloud Computing and Services (MCS)*, pages 53–60, Low Wood Bay, Lake District, United Kingdom, June 2012.

[3] L. Barroso and U. Hölzle. The case for energy-proportional computing. *IEEE Computer*, 40(12):33–37, Dec. 2007.

[4] F. Bellosa. The benefits of event-driven energy accounting in power-sensitive systems. In *ACM SIGOPS European Workshop*, pages 37–42, Kolding, Denmark, Sept. 2000.

[5] Maxwell Technologies BOOSTCAP K2 Datasheet. `http://www.tecategroup.com/capacitors/datasheets/maxwell/K2_series.pdf`.

[6] Maxwell Technologies BOOSTCAP Ultracapacitors Product Guide. `http://www.maxwell.com/products/ultracapacitors/docs/1014627_boostcap_product_guide.pdf`.

[7] M. Brosseau, N. Saunier, K. Le Mouel, and L. Miranda-Moreno. The impact of traffic lights on dangerous pedestrian crossings and violations: a case study in Montreal. In *the 91st Annual Transportation Research Board Meeting*, number 12-0941, Washington, DC, Jan. 2012.

[8] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *USENIX Annual Technical Conf.*, Boston, MA, June 2010.

[9] G. Challen and M. Hempstead. The case for power-agile computing. In *the 13th Workshop on Hot Topics in Operating Systems (HotOS)*, Napa, CA, May 2011.

[10] M. Dong and L. Zhong. Self-constructive high-rate system energy modeling for battery-powered mobile systems. In *the 9th Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys)*, Bethesda, MD, June 2011.

[11] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler. Trio: Enabling sustainable and scalable outdoor wireless sensor network deployments. In *the 5th Intl. Symp. on Information Processing in Sensor Networks (IPSN)*, pages 407–415, Nashville, TN, Apr. 2006.

[12] D. Estrin. Reflections on wireless sensing systems: From ecosystems to human systems. In *IEEE Radio and Wireless Symp.*, pages 1–4, Jan. 2007.

[13] R. Faranda and S. Leva. Energy comparison of MPPT techniques for PV systems. *WSEAS Trans. on Power Systems*, 3(6):446–455, June 2008.

[14] J. Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *the 17th ACM Symp. on Operating Systems Principles (SOSP)*, pages 48–63, Kiawah Island, SC, Dec. 2001.

[15] M. Goldsmith. Try feedforward instead of feedback. *Journal for Quality and Participation*, pages 38–40, 2003.

[16] M. Hassanalieragh, T. Soyata, A. Nadeau, and G. Sharma. Ur-solarcap: An open source intelligent auto-wakeup solar energy harvesting system for supercapacitor based energy buffering.

[17] M. Hassanalieragh, T. Soyata, A. Nadeau, and G. Sharma. Solar-supercapacitor harvesting system design for energy-aware applications. In *the 27th IEEE Intl. System-on-Chip Conf.*, pages 280–285, Las Vegas, NV, Sept. 2014.

[18] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *the 9th Intl. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 93–104, Cambridge, MA, Nov. 2000.

[19] S. Jackson, L. F. Miranda-Moreno, P. St-Aubin, and N. Saunier. A flexible, mobile video camera system and open source video analysis software for road safety and behavioral analysis. *Transportation Research Record: Journal of the Transportation Research Board*, 2365(1):90–98, 2013.

[20] X. Jiang, J. Polastre, and D. Culler. Perpetual environmentally powered sensor networks. In *the 4th Intl. Symp. on Information Processing in Sensor Networks (IPSN)*, Los Angeles, CA, Apr. 2005.

[21] N. Jinrui, W. Zhifu, and R. Qinglian. Simulation and analysis of performance of a pure electric vehicle with a super-capacitor. In *IEEE Vehicle Power and Propulsion Conf.*, pages 1–6, 2006.

[22] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. Power management in energy harvesting sensor networks. *ACM Trans. on Embedded Computing Systems*, 6(4), Sept. 2007.

[23] S. Kim and P. H. Chou. Size and topology optimization for supercapacitor-based sub-watt energy harvesters. *IEEE Trans. on Power Electronics*, 28(4):2068–2080, Apr. 2013.

[24] R. LiKamWa, B. Priyantha, M. Philipose, L. Zhong, and P. Bahl. Energy characterization and optimization of image sensing toward continuous mobile vision. In *the 11th Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys)*, pages 69–82, June 2013.

[25] LTC1624CS8 High-Efficiency SO-8 N-Channel Switching Regulator Controller. `http://cds.linear.com/docs/en/datasheet/1624f.pdf`.

[26] J. C. McCullough, Y. Agarwal, J. Chandrasheka, S. Kuppuswamy, A. C. Snoeren, and R. K. Gupta. Evaluating the effectiveness of model-based power characterization. In *USENIX Annual Technical Conf.*, Portland, OR, June 2011.

[27] C. Min, Y. Lee, C. Yoo, S. Kang, S. Choi, P. Park, I. Hwang, Y. Ju, S. Choi, and J. Song. PowerForecaster: Predicting smartphone power impact of continuous sensing applications at pre-installation time. In *the 13th Intl. Conf. on Embedded Networked Sensor Systems (SenSys)*, pages 31–44, Seoul, South Korea, Nov. 2015.

[28] A. Nadeau, G. Sharma, and T. Soyata. State-of-charge estimation for supercapacitors: A kalman filtering formulation. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2213–2217, Florence, Italy, May 2014.

[29] A. F. O'Connell, J. D. Nichols, and K. U. Karanth. *Camera Traps in Animal Ecology: Methods and Analyses*. Springer, Sept. 2011.

[30] OpenCV. `http://opencv.org/`.

[31] A. Pathak, Y. C. Hu, and M. Zhang. Where is the energy spent inside my app? Fine grained energy accounting on smartphones with Eprof. In *the 7th EuroSys Conf.*, Bern, Switzerland, Apr. 2012.

[32] A. Raghavan, Y. Luo, A. Chandawalla, M. Papaefthymiou, K. P. Pipe, T. F. Wenisch, and M. M. K. Martin. Computational sprinting. In *the 18th Symp. on High Performance Computer Architecture (HPCA)*, Feb. 2012.

[33] C. Renner, F. Meier, and V. Turau. Policies for predictive energy management with supercapacitors. In *8th IEEE Intl. Workshop on Sensor Networks and Systems for Pervasive Computing*, Lugano, Switzerland, Mar. 2012.

[34] B. Ricketts and C. Ton-That. Self-discharge of carbon-based supercapacitors with organic electrolytes. *Journal of Power Sources*, 89(1):64–69, July 2000.

[35] A. Rufer and P. Barrade. A supercapacitor-based energy storage system for elevators with soft commutated interface. *IEEE Trans. on Industry Applications*, 38(5):1151–1159, 2002.

[36] P. St-Aubin, L. Miranda-Moreno, and N. Saunier. A surrogate safety analysis at protected freeway ramps using cross-sectional and before-after video data. In *the 91st Annual Transportation Research Board Meeting*, number 12-2955, Washington, DC, Jan. 2012.

[37] NVidia TEGRA3 Mobile Processor. `http://www.nvidia.com/object/tegra-3-processor.html`.

[38] Texas Instruments Incorporated. Designing DC/DC converters based on SEPIC technology. `http://www.ti.com/lit/an/slyt309/slyt309.pdf`, 2013.

[39] D. Wang, C. Ren, A. Sivasubramaniam, B. Urgaonkar, and H. Fathy. Energy storage in datacenters: What, where, and how much? In *ACM SIGMETRICS Conf.*, London, UK, June 2012.

[40] A. S. Weddell, G. V. Merrett, T. J. Kazmierski, and B. M. Al-Hashimi. Accurate supercapacitor modeling for energy harvesting wireless sensor nodes. *IEEE Trans. on Circuits and Systems II: Express Briefs*, 58(12):911–915, Dec. 2011.

[41] P. F. Williams. Street smarts: How intelligent transportation systems save money, lives and the environment. Technical report, ACS Transportation Solutions Group, Xerox, Feb. 2009.

[42] P. J. Wolff, E. J. Heske, R. L. Schooley, and C. A. Taylor. Predation risk for crayfish in streams within a human-dominated landscape. `www.midwestfw.org/documents/WildlifePosters.pdf`, Dec. 2012. Poster session in the 73rd Midwest Fish and Wildlife Conference.

[43] J. Wu, J. Wang, K. Li, H. Zhou, Q. Lv, L. Shang, and Y. Sun. Large-scale energy storage system design and optimization for emerging electric-drive vehicles. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(3):325–338, Mar. 2013.

[44] S. Zangenehpour, L. F. Miranda-Moreno, and N. Saunier. Impact of bicycle boxes on safety of cyclists: A case study in Montreal. In *the 92nd Annual Transportation Research Board Meeting*, number 13-2909, Washington, DC, Jan. 2013.

[45] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Intl. Conf. on Hardware-Software Codesign and System Synthesis (CODES+ISSS)*, Scottsdale, AZ, Oct. 2010.

[46] Y. Zhang and H. Yang. Modeling and characterization of supercapacitors for wireless sensor network applications. *Journal of Power Sources*, 196(20):4128–4135, 2011.

[47] T. Zhu, Z. Zhong, Y. Gu, T. He, and Z.-L. Zhang. Leakage-aware energy synchronization for wireless sensor networks. In *the 7th Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys)*, pages 319–332, Kraków, Poland, June 2009.

[48] ZoneMinder. `www.zoneminder.com`.

[49] L. Zubieta and R. Bonert. Characterization of double-layer capacitors for power electronics applications. *IEEE Trans. Ind. Appl.*, 36(1):199–205, 2000.