Battery- and Aging-Aware Embedded Control Systems for Electric Vehicles

Wanli Chang*, Alma Pröbstl†, Dip Goswami‡, Majid Zamani†, Samarjit Chakraborty†
*TUM CREATE, Singapore †TU München, Germany ‡Eindhoven University of Technology, Netherlands
wanli.chang@tum-create.edu.sg

***Abstract*—In this paper, for the first time, we propose a battery- and aging-aware optimization framework for embedded control systems design in electric vehicles (EVs). Performance and reliability of an EV are influenced by feedback control loops implemented into in-vehicle electrical/electronic (E/E) architecture. In this context, we consider the following design aspects of an EV: (i) battery usage; (ii) processor aging of the in-vehicle embedded platform. In this work, we propose a design optimization framework for embedded controllers with gradient-based and stochastic methods taking into account quality of control (QoC), battery usage and processor aging. First, we obtain a Pareto front between QoC and battery usage utilizing the optimization framework. Well-distributed non-dominated solutions are achieved by solving a constrained bi-objective optimization problem. In general, QoC of a control loop highly depends on the sampling period. When the processor ages, on-chip monitors could be used to measure the delay of the critical path, based on which, the processor operating frequency is reduced to ensure correct functioning. As a result, the sampling period gets longer opening up the possibility of QoC deterioration, which is highly undesirable for safety-critical applications in EVs. Utilizing the proposed framework, we take into account the effect of processor aging by re-optimizing the controller design with the prolonged sampling period resulting from processor aging. We illustrate the approach considering electric motor control in EVs. Our experimental results show that the effect of processor aging on QoC deterioration can be mitigated by controller re-optimization with a slight compromise on battery usage.**

***Keywords*-embedded control system; processor aging; battery rate capacity effect; electric vehicle;**

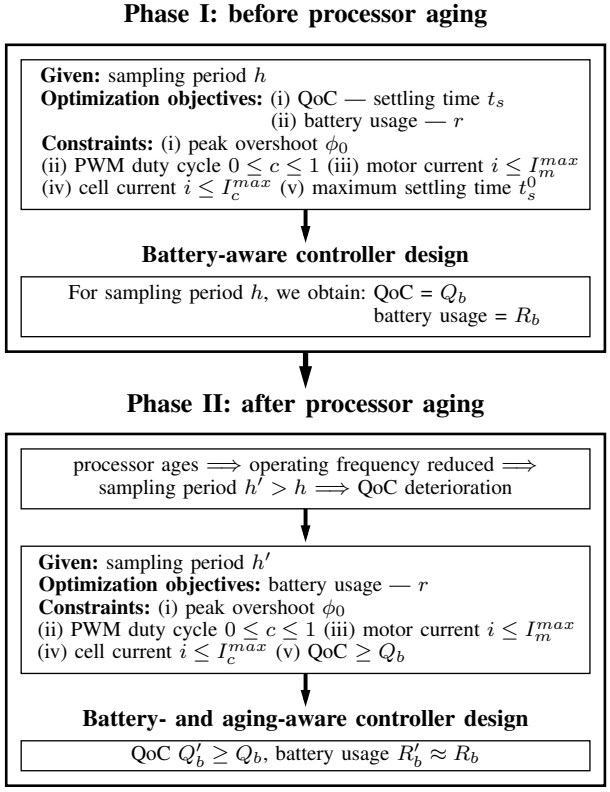

Figure 1. The design flow of embedded control systems for EVs

I. Introduction

A reduced emission, independence from fossil fuels and improvement of energy conversion efficiency have made electric vehicles (EVs) a potential alternative of conventional vehicles with internal combustion engines. Design of underlying embedded control loops such as electric motor control, anti-lock braking control, yaw stabilization, adaptive cruise control, and battery management plays a crucial role in EVs. They are usually evaluated by a number of Quality-of-Control (QoC) indices. One common QoC metric is settling time. In order to ensure vehicle performance and reliability, the design also needs to take into account a number of application-specific issues such as battery usage and processor aging. Battery is the key component influencing the performance of an EV. As the integrated circuit fabrication technology has progressed, processors become more and more susceptible to aging [1]. In order to ensure correct functioning, the processor operating frequency has to be reduced, which could potentially worsen QoC and compromise vehicle reliability. Our focus in this work is to propose a novel battery- and aging-aware design framework for embedded control systems in EVs. The entire flow is illustrated in Figure 1.

For all practical purposes, a longer driving range is desired in EVs to increase their usability. A battery pack with large *capacity* is needed to offer a long driving range. However, with larger capacity, the battery weight also increases leading to higher energy consumption. Moreover, the capacity is restricted by the space that can be allocated to the battery pack in EVs. One potential solution to the above problem is to design the controller in such a way that the energy consumption of the control task can be minimized.

All off-the-shelf battery packs are labeled with a nominal capacity. However, due to the rate capacity effect, the full charge capacity (FCC) of a battery pack, which is defined to be the amount of electric charges that can be delivered from the battery after it is fully charged, actually varies with different discharging current profiles [2], [3]. Generally

1052-8725/14 $31.00 © 2014 IEEE
DOI 10.1109/RTSS.2014.24

speaking, larger discharging current tends to reduce the FCC. For most common lithium-ion batteries in the market, the capacity could potentially get significantly compromised if the rate capacity effect is not properly considered in the control systems design. In this work, we introduce an optimization framework considering QoC as one design objective and battery usage as the other. We quantify the battery usage by the number of times the control system can reach a steady state after a disturbance occurs powered up by a fully charged battery pack. In order to maximize the battery usage, the energy consumption of the control task should be small and the battery FCC should be increased by generating a battery-friendly discharging current profile.

In this context, the other important design aspect is processor aging. As a processor ages, the switching time of its transistors increases, resulting in longer path delays. On-chip monitors could be used to measure the delay of the critical path. It always has to be guaranteed that the signal transmission can be complete along any path within one clock cycle [4]. Therefore, the processor operating frequency is reduced based on the new critical path delay. On the other hand, a shorter sampling period can potentially provide a better QoC. Therefore, with a smaller processor operating frequency, the sampling period increases and QoC gets deteriorated, which is dangerous and thus highly unwanted for safety-critical applications such as electric motor control in EVs. To deal with the above situation, we propose to re-optimize the controller with the longer sampling period, which results from processor aging.

Contributions: As shown in Figure 1, the entire design flow of battery- and aging-aware embedded control systems in EVs is divided into two phases. In Phase I, before the processor ages, we propose an optimization framework with both QoC and battery usage considered as design objectives. With gradient-based and stochastic methods implemented, this battery-aware controller design gives us a Pareto front of well-distributed and non-dominated solutions. The trade-off between these two objectives is explored. In Phase II, after the processor ages, QoC is found to get degraded if the controller design does not change. The same optimization framework is used with slight modification. The processor aging effect is mitigated by this battery- and aging-aware controller design in the way that QoC does not get deteriorated with an inconsiderable compromise on battery usage.

Organization: The remainder of this paper is organized as follows: Section II gives an overview of related work in embedded control systems design, battery rate capacity effect and processor aging. Section III describes feedback control loops and the electric motor control application, in particular. Derivation of QoC as one design objective is explained. In Section IV, two design aspects for embedded control systems in EVs are discussed. Battery usage is introduced as the other design objective with the rate capacity effect taken into account. Processor aging and its influence on the control system sampling period are analyzed. Then we present our optimization framework and flow in Section V. Experimental results are reported in Section VI and, finally, Section VII concludes the paper.

II. Related Work

Our work is closely related to choices of sampling periods in control loops, battery rate capacity effect and processor aging. Multirate sampling schemes are well-known techniques in control theory to deal with the sampling periods [5], [6]. These works consider the setups where sensing devices have a different sampling rate from actuating devices, and the focus is to ensure stability of the overall system under such multirate sampling. An appropriate choice of a sampling period is important to be considered while designing an embedded controller. Some recent works in this direction studied the choice of sampling periods both for networked architectures [7] and single-processor implementation platforms [8], [9], [10]. However, none of the works have directly investigated the relationship of the control system with properties of the input energy source (i.e., battery) and effects caused by underlying processing platform aging, which are two important aspects in embedded control systems design for EVs.

There has been extensive work to model the battery behavior. One important issue to address is the rate capacity effect that the capacity of a battery cell varies with different discharging current profiles. In general, researchers have developed three types of battery models to characterize this effect. An electrochemical model is based on chemical processes taking place within the battery [11]. These models describe the battery processes in great detail, which makes them the most accurate among all types of models. However, highly detailed description makes the models complex and difficult to configure. Electrical-circuit models use circuit elements to simulate battery behavior. The first such model was proposed in [12]. They are simpler than the electrochemical models and thus computationally less expensive. However, it still takes considerable efforts to configure the electrical-circuit models. Especially the lookup tables used in the model require lots of experimental data on the battery behavior. An analytical model describes the battery at a higher level of abstraction than the other two types. The major properties of the battery are modeled by a couple of equations, which makes this type of model much easier to use. In this work, we use extended Peukert's law presented in [13] to characterize battery capacity rate effect. With carefully chosen parameters based on experimental data, reasonable accuracy of about $10\,\%$ error can be achieved.

As a processor ages, the transistor switching time is increased and the path delay gets prolonged. Signals may not be able to go through some of the paths within one clock cycle, which results in timing constraint violation and false calculation. This issue can be handled by safety margins, supply voltage regulation or clock frequency scaling [14]. Currently, the most widely-used method is implementing a frequency or voltage guard band. The processor is operated at a lower frequency or higher voltage that meets worst-case path delays. It is noted that a higher supply voltage makes transistor switching faster. In this way, as the processor ages and gets slower, the system does not have to be changed. However, the guard band can be quite big, leading to a passive design and a waste of resources [15]. In cost-sensitive domains like EVs, there is a strong motivation to

make full use of resources and minimize the guard band, while reliable functionality is maintained. Instead of having a fixed setting that deals with worst-case conditions, some newer approaches suggest changing settings dynamically. By monitoring the critical path, delays are detected and the supply voltage of the processor is adjusted accordingly [16]. These approaches are effective but can be difficult to implement. The supply voltage is usually limited by the maximum allowable input current in the circuit, cooling constraints and other temperature-dependent reliability issues, which are sensitive in the automotive domain. In this work, we use autonomous frequency scaling to deal with processor aging [14]. On-chip monitors could be used to watch the critical path delay, based on which, the processor operating frequency is adjusted by a frequency synthesis circuit to ensure that signals can go through all paths within one clock cycle. Timing constraints are respected and functions are guaranteed to be correct.

III. Feedback Control Applications

In this section, we first describe the feedback control application considered in this work and several background concepts. It is then explained how a controller can be designed and the objective QoC is derived. Finally, we specifically present system modeling of the electric motor control application in EVs.

A. Basic concepts

Plant dynamics: A control scheme is responsible for controlling a plant or dynamical system. In this work, we consider linear single-input single-output (SISO) systems,

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t), \\ y(t) &= \mathbf{C}\mathbf{x}(t), \end{aligned} \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector, $\dot{\mathbf{x}}(t)$ is the derivative of $\mathbf{x}(t)$ with respect to time, $y(t)$ is the output, and $u(t)$ is the control input to the system. Matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are of appropriate dimensions. In a state-feedback control algorithm, the control input $u(t)$ is computed utilizing the system states $\mathbf{x}(t)$ as feedback signals. The computed $u(t)$ is then applied to the plant, which is expected to achieve the desired behavior.

Discretized dynamics: In an embedded implementation platform, the overall loop performs three operations — measurement, computation, and actuation. In measurement, the system states $\mathbf{x}(t)$ are measured with sensors and sent to the controller. In computation, the control input $u(t)$ is calculated. In actuation, the computed $u(t)$ is applied to the plant. Generally, these three operations form one sampling period. When the sampling period is constant, the continuous-time system in (1) can be transformed into a discrete-time system,

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{A}_d\mathbf{x}[k] + \mathbf{B}_d u[k], \\ y[k] &= \mathbf{C}_d\mathbf{x}[k], \end{aligned} \tag{2}$$

where sampling instants are $\{t_k | k \in \mathbb{N}\}$ and the sampling period is $t_{k+1} - t_k = h$. It is noted that $\mathbf{x}[k]$, $u[k]$ and $y[k]$ are the values of $\mathbf{x}(t)$, $u(t)$ and $y(t)$ at $t = t_k$ and

$$\mathbf{A}_d = e^{\mathbf{A}h},\ \mathbf{B}_d = \int_0^h (e^{\mathbf{A}t}dt) \times \mathbf{B},\ \mathbf{C}_d = \mathbf{C}. \tag{3}$$

Feedback controller: One common control objective is to make $y[k] \rightarrow y_{ref}$ as soon as possible, where y_{ref} is the desirable value for $y[k]$ to achieve. Towards this and other application-specific control objectives, we need to design $u[k]$ utilizing the states $\mathbf{x}[k]$. This is then a state-feedback controller with a general structure as follows,

$$u[k] = \mathbf{K} \times \mathbf{x}[k] + F \times y_{ref}, \tag{4}$$

where $\mathbf{K}$ is the feedback gain vector and F is the feedforward gain.

Closed-loop system: With a feedback controller as shown in (4), the system closed-loop dynamics in (2) becomes,

$$\mathbf{x}[k+1] = (\mathbf{A}_d + \mathbf{B}_d\mathbf{K})\mathbf{x}[k] + \mathbf{B}_d F y_{ref}. \tag{5}$$

The feedback gain vector $\mathbf{K}$ is designed to stabilize the closed-loop system matrix $(\mathbf{A}_d + \mathbf{B}_d\mathbf{K})$. In other words, $(\mathbf{A}_d + \mathbf{B}_d\mathbf{K})$ should have all eigenvalues with absolute values of less than unity. A commonly used technique to compute $\mathbf{K}$ is pole-placement. In pole-placement methods like Ackermann's formula and Linear Quadratic Regulator (LQR) design [17], the feedback gain vector $\mathbf{K}$ is designed by placing the eigenvalues (i.e., poles) of $(\mathbf{A}_d + \mathbf{B}_d\mathbf{K})$ at the desired location p. Referring to the above setting, we have

$$\forall i,\ 0 \leq |p_i| \leq 1. \tag{6}$$

The number of poles is the same as the number of states. Different choices of poles p result in different QoC. The static feedforward gain F is designed to achieve $y[k] \rightarrow y_{ref}$ and computed by

$$F = 1/(\mathbf{C}_d(\mathbf{I} - \mathbf{A}_d - \mathbf{B}_d \times \mathbf{K})^{-1}\mathbf{B}_d), \tag{7}$$

where $\mathbf{I}$ is an n-dimensional identity matrix.

Quality of Control: We use settling time as a QoC metric. It is defined to be the time duration t_s necessary for $y[k]$ to reach and stay in the ϕ_e range of y_{ref}.

$$\forall kh \geq t_s,\ |y[k] - y_{ref}| \leq \phi_e \times y_{ref}, \tag{8}$$

where ϕ_e is the settling time threshold. t_s is considered as one design objective in the embedded control systems design of this work. Shorter t_s indicates better QoC. A number of control applications in EVs impose stringent requirements on the maximum allowed settling time, i.e., $t_s \leq t_s^0$ [18]. In most cases, these requirements are safety-critical.

Further, we consider peak overshoot as another constraint which is defined as follows,

$$y_{max} - y_{ref} \leq \phi_0 \times y_{ref}, \tag{9}$$

where y_{max} is the maximum value of the system output. The overshoot is important in the automotive context to take into account the user comfort level. In the application of electric motor control, we consider $\phi_e = 2\%$ and $\phi_0 = 5\%$.

Input saturation: In general, the input signal $u[k]$ is constrained by an upper limit U_{max}, e.g., voltage or current

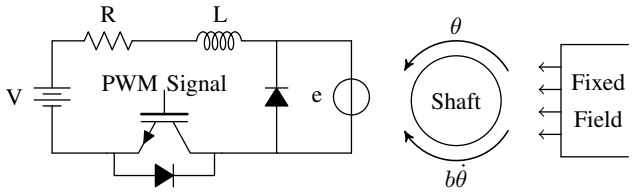

Figure 2. A diagram of a DC motor with the armature circuit powered up by a battery pack

limit of the input energy source. Therefore, the controller needs to be designed such that the maximum value of $u[k]$ does not exceed this limit, i.e., $u[k] \leq U_{max}$.

B. Electric motor control

Electric motor control is a key function in EVs. As shown in Figure 2, we consider a DC motor running in the *speed control* mode. The controller is supposed to operate the motor at various speeds according to driver input or environmental requirements. V is the DC voltage provided by the battery pack. R and L are resistance and inductance in the armature circuit. e is the back electromotive force (EMF) from the motor. The insulated gate bipolar transistor (IGBT) works as a switch controlled by pulse-width modulation (PWM) signals at the gate. When the switch is on, V is applied to the armature circuit. When the switch is off, the diode flows out remaining current in the motor and thus the applied voltage is equivalent to zero. Periodic PWM signals are shown in Figure 3, where the duty cycle c is calculated as

$$c = \frac{t_{on}}{t_{period}}, \tag{10}$$

and the effective voltage applied in the armature circuit is

$$V_{eff} = cV. \tag{11}$$

We can clearly see that V_{eff} is adjustable by controlling PWM signals.

In general, the torque T generated by a DC motor is proportional to the armature current i and the strength of the magnetic field. We assume the magnetic field to be constant and thus the torque is calculated as

$$T = K_t i, \tag{12}$$

where K_t is the motor torque constant. We denote the angular position of the motor to be θ. The angular velocity and acceleration are then $\dot{\theta}$ and $\ddot{\theta}$, respectively. The back EMF is proportional to the angular velocity of the shaft by a constant factor K_e as follows,

$$e = K_e \dot{\theta}. \tag{13}$$

A viscous friction model is assumed and the friction torque is proportional to the shaft angular velocity $\dot{\theta}$ by a factor of b. Now we can derive the following governing equations based on Newton's second law and Kirchhoff's law [19],

$$\begin{aligned} J\ddot{\theta} + b\dot{\theta} &= K_t i, \\ L\frac{di}{dt} + Ri &= cV - K_e\dot{\theta}, \end{aligned} \tag{14}$$

where J is the moment of inertia of the motor.

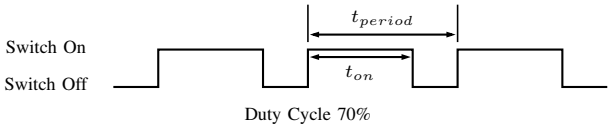

Figure 3. Pulse-width modulation control signals in the armature circuit to adjust the DC voltage applied to the rotor

The state-space system modeling as in (1) becomes

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} &= \begin{bmatrix} -\frac{b}{J} & \frac{K_t}{J} \\ -\frac{K_e}{L} & -\frac{R}{L} \end{bmatrix}\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{V}{L} \end{bmatrix} c, \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}. \end{aligned} \tag{15}$$

The states are the angular velocity of the motor $\dot{\theta}$ and the armature current i. The starting state is denoted as $[\dot{\theta}_0\ i_0]^T$. The control input is the duty cycle c and the system output is $\dot{\theta}$. The control task is to make $\dot{\theta}$ approach a certain value $\dot{\theta}_{ref}$. As determined by the matrix dimension, there are two eigenvalues to design. It is clear that the constraint on the control input is

$$0 \leq c \leq 1. \tag{16}$$

Another constraint is on the largest current that can be sustained by wires in the motor, i.e.,

$$i \leq I_m^{max}. \tag{17}$$

IV. Design Aspects of Electric Vehicles

In this section, we discuss two important design aspects of EVs — battery usage and processor aging. Besides t_s as one design objective for QoC, we make r the other design objective to quantify battery usage. r is defined to be the number of times the control system can reach a steady state after a disturbance occurs with a fully charged battery pack, taking account of the rate capacity effect. Then processor aging and its influence on QoC are analyzed.

A. Battery Rate Capacity Effect

The battery pack is one of the most important components in EVs. Battery capacity is the major factor determining the driving range of an EV. A natural objective of embedded control systems design for EVs is to minimize the energy consumption of each control task execution. However, this is not enough to thoroughly and accurately consider the energy impact from control systems design. Due to battery rate capacity effect, the FCC of a battery pack varies depending on discharging current profiles. This effect needs to be considered in embedded control systems design for EVs since different controller designs result in different current profiles. The rate capacity effect is described by extended Peukert's law [13] as shown below,

$$L_t = \frac{a}{\left(\frac{\sum_{k=1}^{n} I_k(t'_{k+1} - t'_k)}{L_t}\right)^d}, \tag{18}$$

which is able to handle non-constant loads. $t'_1 = 0$ is the starting time stamp and $L_t = t'_{n+1}$ is the total duration that the battery can be used and divided into n slots. In this particular work, each slot is equal to one sampling period of

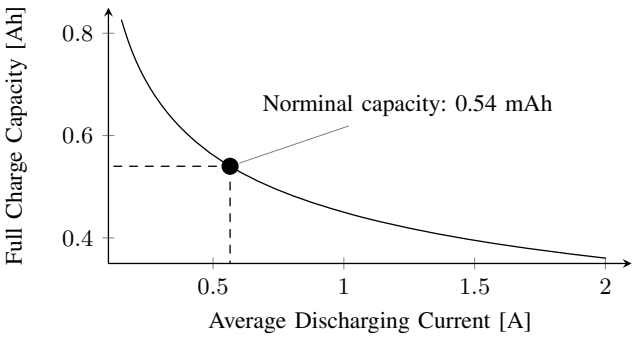

Figure 4. Relationship between battery FCC and average discharging current

the control system. For the kth time slot, I_k is the average current for the sampling period from t_k to t_{k+1}. a and d are determined by experimental data. It is noted that this is an approximation of the battery rate capacity effect by considering the average discharging current. As discussed in Section II, more accurate methods can be used as well, and the entire framework presented in this paper still holds. The battery capacity under the discharging current profile $\{(I_1, t_1), (I_2, t_2), \ldots, (I_n, t_n)\}$ is

$$Q_{FCC} = \sum_{k=1}^{n} I_k(t_{k+1} - t_k). \tag{19}$$

Ideally, both Q_{FCC} and a are equal to the nominal battery capacity Q_{nom} and $d = 1$.

In this work, we use the lithium-ion rechargeable batteries with lithium cobalt oxide cathode and graphite anode produced by Sony (UP383562). The operating voltage of each cell is $3.7\,V$. One constraint on the battery pack is that the current drawn from each cell cannot exceed a certain value I_c^{max}. From the experimental data in [20], a and d can be calculated as 0.45 and 1.32, respectively. The relationship between the FCC of a battery cell and the average discharging current is shown in Figure 4. The nominal capacity labeled with the battery is $0.54\,Ah$.

Besides t_s as a design objective to quantify QoC, we propose battery usage as the other objective in the embedded control systems design for EVs, which is characterized by the number of times r the control system can reach a steady state after a disturbance occurs with a fully charged battery pack. In order to maximize r, (i) the energy consumption of each single control execution has to be small, and (ii) the average discharging current is desirable to be small to improve the FCC. Assuming that the battery only powers the electric motor control task, based on control system description in Section III and (18), r can be calculated as

$$r = \frac{a}{\left(\frac{\sum_{k=1}^{n_{sp}} I_k(t_{k+1}-t_k)}{t_s}\right)^d \times t_s}, \tag{20}$$

where t_s is the time taken to complete one single control task, i.e., settling time, as discussed in Section III. $t_{k+1} - t_k$ is the sampling period of the control system, i.e., h. These time slots are constant since we deal with control systems with constant sampling period. n_{sp} is the total number of sampling periods in each execution. As presented in Section III, when the controller design is decided, from experiments we can get the value of t_s and the supplied current profile, based on which the sum of current $\sum_{k=1}^{n_{sp}} I_k(t_{k+1} - t_k)$ can be derived. In general, with different controller designs, i.e., eigenvalue selections, these two objectives t_s and r take different values. The relationship between these two objectives is not explicit since the sum of current also varies depending on how the controller is designed. Therefore, it is necessary to develop an optimization technique to optimize both t_s and r while all constraints must be satisfied.

B. Processor Aging in Embedded Control Systems

Electronics plays a major role in embedded control systems of modern vehicles. A processor is the key part ensuring correct functioning and timing behavior in control. As the transistor size keeps shrinking, processors are becoming more and more susceptible to aging, which could get QoC degraded. This is dangerous and highly undesirable in safety-critical applications like EVs. The main transistor aging mechanisms are Hot Carrier Injection (HCI) and Negative Bias Temperature Instability (NBTI) [21]. They cause degradation in the electrical characteristics of transistors, such as a shift of the threshold voltage [22]. As a result, the switching times of transistors are prolonged.

There are many paths for digital signal transmission in a processor and each path consists of a number of transistors. The time that a signal takes to travel through the path is called path delay. As transistors require more time for switching, the path delay is increased [4]. It has to be ensured that the signal transmission along any path can be completed within one clock cycle. Therefore, the operating frequency of the processor is determined by the worst-case path delay, which is the path with the longest delay and called the critical path. With the above analysis, we know that as a processor ages, the signal may not be able to travel through certain paths within one clock cycle if the operating frequency remains fixed. This could produce incorrect functional results, that cannot be tolerated by safety-critical applications like motor control in EVs.

As discussed in Section II, one appropriate way to keep functions correct in the domain of EVs is reducing the operating frequency. Before everything, we need to know the extend of aging. An on-chip aging monitor could be implemented to watch the delay of the critical path. Shrinking transistor dimensions make larger within-die and die-to-die variations [23]. The static timing analysis at the circuit level is not able to precisely determine which path among all is the critical one in modern processors. However, we can identify all critical path candidates, which are a subset of all paths in the processor that can become the critical one during the runtime. To avoid interference on real functions run on the processor, Critical Path Replicas (CPRs), which are replicas of these candidates, are fabricated on the chip [23]. The aging monitor then sends signals to CPRs to get the current critical path delay, based on which it is known whether the operating frequency needs to be reduced and if so, by how much. A frequency generator changes the operating frequency accordingly, if necessary.

When the processor operating frequency is decreased, the execution time of control algorithms gets longer. The sampling period as described in Section III is mainly constrained by the control algorithm worst-case execution time (WCET). Therefore, the sampling period of the control system is increased. It is possible to avoid the aging effect by inserting a safety margin between the WCET and the sampling period. However, this is a waste of resources and not desirable for cost-sensitive applications like EVs. In general, a shorter sampling period means more frequent response from the controller and thus potentially better QoC. When the increased sampling period due to processor aging is fed into the control system presented in Section III, we can obtain the results showing how QoC changes.

Based on the method presented in [14], we are able to estimate how much processor aging can be expected in EVs. The processor only ages when it is used, i.e., in the non-idle state. Taking an electric taxi as an example, the drivers take shifts and the car is driven approximately 20 hours every day. The processor is always in the non-idle state while the car is driven. After 4 years of use, the overall operation time of the processor is approximately $\frac{5}{6} \times 4 \approx 3.33$ years. According to the results presented in [14], the processor is roughly $10\,\%$ slower.

V. Design optimization framework

As discussed in Section III and IV, we consider three design aspects – QoC, battery usage and processor aging. In this section, we first present the optimization framework for battery-aware controller design, which is Phase I as illustrated in Figure 1. This is a constrained bi-objective optimization problem with QoC and battery usage as objectives. Both gradient-based and stochastic methods are used to solve it. They complement each other with their respective advantages to obtain well-distributed non-dominated Pareto points. An algorithm is proposed to further improve the distribution quality. The trade-off between QoC Q_b and battery usage R_b can then be explored. Afterwards, we slightly modify the framework for Phase II, i.e., battery- and aging-aware controller design to ensure that QoC keeps not getting degraded, i.e., $Q_b' \geq Q_b$.

A. Optimization for Phase I: battery-aware controller design

Problem formulation: In this battery-aware controller design optimization problem, the two closed-loop eigenvalues (poles) p_1 and p_2 are decision variables to tune. We restrict the space for the eigenvalues of the closed-loop system in the real positive plane — which is the case in most of the real-life design problems. This is particularly acceptable considering the possible oscillation with complex eigenvalues in speed control of EVs. Therefore, based on (6) we have

$$\forall i \in \{1, 2\},\ 0 \leq p_i \leq 1. \tag{21}$$

It can be seen that the design space is continuous. As discussed before, the two objectives to optimize are

- the settling time of the control task t_s and
- the number of times r the control system can reach a steady state after a disturbance occurs with a fully charged battery pack.

t_s should be minimized and r should be maximized. Usually an optimization technique takes objectives either to minimize or maximize but not both. Therefore, in this work, we minimize $f_1 = t_s$ and $f_2 = -r$.

There are several constraints to be satisfied. As shown in (16), the control input, i.e., the duty cycle of the PWM control signals c is constrained as $0 \leq c \leq 1$. As shown in (17), the current in the motor cannot exceed the maximum allowed current, i.e., I_m^{max}. The current drawn from each cell cannot be larger than I_c^{max} as explained in Section IV-A. For automotive control applications, timing is critical. It is necessary to guarantee that each control task meets the settling time requirement, leading to the constraint $t_s \leq t_s^0$ as discussed in Section III. The last constraint is related to overshoot and explained in (9).

Optimization goals: In solving a constrained bi-objective optimization problem, there are typically two goals to achieve:

- the solution point is on the Pareto surface and
- there is a good distribution of solution points.

The first goal ensures that the solution point is not dominated by any other point. In other words, no other point is better than the solution in both objectives. The second goal gives designers more freedom to choose solutions under different circumstances. In this controller design optimization problem with significant non-linearity and non-convexity, the design space is continuous with infinite design choices. There is no relationship between objectives and decision variables that can be explicitly formulated. Therefore, it is a challenging task to achieve both optimization goals. There are generally two types of heuristic techniques to solve such a constrained bi-objective optimization problem: gradient-based and stochastic methods [24]. They have different advantages and disadvantages.

Sequential Quadratic Programming (SQP): SQP is a popular gradient-based optimization technique. Since it is originally meant for single-objective optimization problems, we need to first convert the two objectives into one. One common way to do so is constructing a new objective f_0 to be minimized with a weighted sum of f_1 and f_2 as shown below,

$$f_0 = w_1 \times f_1 + w_2 \times f_2, \tag{22}$$

where w_1 and w_2 are weights of the two objectives f_1 and f_2 and

$$w_1 + w_2 = 1. \tag{23}$$

After conversion into a single-objective optimization problem, the search process begins with a chosen starting point. From this starting point, SQP finds a sequence of points, hopefully closer and closer to the optimal point until either the optimal point is reached or termination criteria are satisfied. The question is then how to find the next point, which is supposed to be better than the current point in terms of the objective value f_0. We need to know the direction and how far to go along this direction. It is not easy to compute them for a problem with significant non-linearity. Therefore, an approximated local quadratic model is built around this

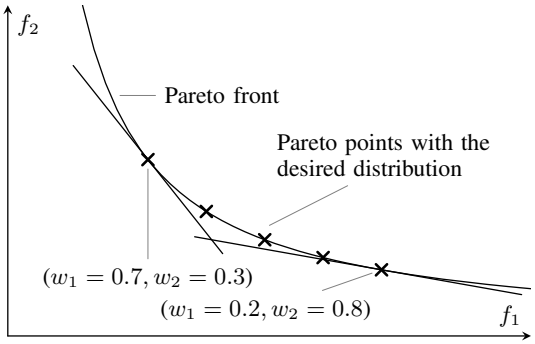

Figure 5. Illustration of how SQP locates Pareto points with various pairs of weights

current point as

$$f_0(\mathbf{x}_p) = \frac{1}{2}\mathbf{x}_p^T\mathbf{H}\mathbf{x}_p + \mathbf{c}_q^T\mathbf{x}_p, \tag{24}$$

where $\mathbf{H}$ is the symmetric $n \times n$ Hessian matrix at this point and $\mathbf{x}_p$ is the location of the current point in the variable space. Both $\mathbf{c}_q$ and $\mathbf{x}_p$ are column vectors with n elements.

The search direction can be calculated in various ways. In this work, we use the direction of steepest descent $\mathbf{d}_s$ with potentially the best local improvement in the objective value f_0, which is calculated as the opposite of the gradient,

$$\mathbf{d}_s = -\frac{df_0}{d\mathbf{x}_p} = -\mathbf{x}_p^T\mathbf{H} - \mathbf{c}_q^T. \tag{25}$$

After calculating the search direction, the next step is to determine how far to go in this direction. Assuming that Δ is the radius of the region where we have confidence in the accuracy of the approximated local quadratic model, a list of steps $\{\frac{\Delta}{n_{ss}}, \frac{2\Delta}{n_{ss}}, \ldots, \Delta\}$ are tried, where n_{ss} is the total number of steps and can be decided by the designer. The step size with the best objective value f_0 and all constraints fulfilled is chosen. Then a new quadratic model is established around this point and the search process is iterated. When no step size gives improvement in f_0, the optimization algorithm is terminated. The other termination criterion is that the maximum number of iterations set by the designer is exceeded. It is noted that all constraints are respected throughout the entire optimization flow.

If SQP is able to reach global optima, when f_0 in (22) is minimized as the objective, the solution point must be on the Pareto surface. This can be proven by contradiction. If there is another point $\mathbf{x}'$, which dominates the solution point $\mathbf{x}_0$, then

$$f_1(\mathbf{x}') < f_1(\mathbf{x}_0),\ f_2(\mathbf{x}') < f_2(\mathbf{x}_0). \tag{26}$$

From (22), we have

$$f_0(\mathbf{x}') < f_0(\mathbf{x}_0). \tag{27}$$

This contradicts that $\mathbf{x}_0$ is the global optimum for minimizing f_0. However, SQP is good at finding local optima and also easily gets trapped by them, which makes it lose track of global optima. A common way to improve the global awareness of SQP is taking multiple starting points to explore different regions. But it is often a challenging task to select an appropriate list of starting points covering the entire objective space. In practice, starting point selection is usually on the random base and thus the objective space is only partially searched.

SQP is also weak in locating a good distribution of Pareto points. Different pairs of weights (w_1, w_2) find different Pareto points. For example, optimizing f_0 constructed by $(w_1 = 1, w_2 = 0)$ and $(w_1 = 0, w_2 = 1)$ finds the solution optimizing f_1 and f_2, respectively. And $(w_1 = 0.5, w_2 = 0.5)$ corresponds to the point with equal emphasis on both objectives. However, a set of equally distant pairs of weights does not necessarily produce a set of equally distant Pareto points. As shown in Figure 5, when applying a pair (w_1, w_2), the straight line with the gradient of $-\frac{w_1}{w_2}$ starts from the lower-left corner and approaches the Pareto front. The first intercepting point between the straight line and the Pareto front is then the Pareto point corresponding to this pair (w_1, w_2). Given the convex Pareto front in Figure 5, theoretically, an appropriate set of weight pairs is able to locate a good distribution of Pareto points, but it is often difficult to identify such a set since the information regarding the Pareto front is always missing. It becomes especially difficult when the gradient change along the Pareto surface is small. Besides, Pareto points in the non-convex region of the Pareto front cannot be located by SQP at all.

Non-dominated sorting genetic algorithm (NSGA): Stochastic methods have also been developed to solve such a constrained bi-objective optimization problem. In this work, we consider the popular NSGA [25]. At first, an initial population is generated and serves as parents. Offsprings are then produced with crossover and mutation. The crossover function tries to keep the good genes of parents, which in this case means that the offsprings are close to parents in the decision variable space. The mutation function deviates from parents and produces offsprings in unexplored regions of the objective space. Elitism is implemented for environmental selection, so that the next generation is selected among both parents and offsprings. This not only speeds up convergence but also ensures that good solutions will not be lost once they are found. There are two termination conditions whether the population has converged and whether the maximum allowed number of generations has been exceeded.

In selection, all parents and offsprings are sorted and ranked by its level of domination. For each solution, if it is not dominated by any other solution, then it is given a rank of 1. All solutions with the rank of 1 form the first front. Then the first front is removed and the second front is formed with non-dominated solutions among the rest. All solutions on the second front have a rank of 2. This process is iterated until all solutions are assigned a rank. The new generation is filled in the way that solutions with lower ranks have priorities. This non-dominated sorting feature in environmental selection favors those solutions on or close to the Pareto front. It is noted that we keep all constraints fulfilled at all time.

The advantage of NSGA over SQP lies in its stochastic nature. More of the objective space can be explored, so that it is very likely that a good distribution of Pareto points can be selected from the final converged population. NSGA

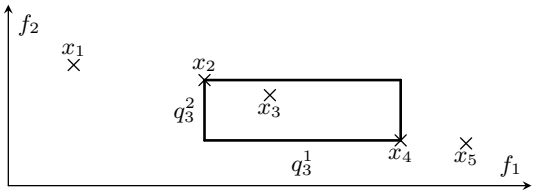

Figure 6. Illustration of the crowding distance calculation for Pareto points distribution quality quantification

is not easily trapped by local optima and thus has better global awareness. This is also its weakness that converged solutions are not guaranteed to be local optima. Considering that both gradient-based and stochastic optimization methods have different advantages, we implement both of them in this work and combine the solutions found by them. The overall runtime is in minutes and scalability is not an issue because the number of objectives is fixed to be two and the controller design is an offline task.

Distribution quality metric: There are various ways to quantify the distribution quality for a set of Pareto points. In this problem of embedded controller design for EVs, the distribution in the objective space instead of decision variable space is investigated since it is important to see the trade-off between QoC and battery usage. We modify the popular method of crowding distance calculation described in [25] to suit this work. As illustrated in Figure 6, assuming that there are two objectives $\{f_1, f_2\}$ and n solution points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ ordered by the value of either objective, for each point $\mathbf{x}_i$, $i \in \{1, 2, \ldots, n\}$, that is not at the end of this point sequence, the crowding distance of $\mathbf{x}_i$ in terms of the objective f_k, $k \in \{1, 2\}$ can be calculated as

$$q_i^k = |f_k(\mathbf{x}_{i+1}) - f_k(\mathbf{x}_{i-1})|, \tag{28}$$

where $\mathbf{x}_{i+1}$ and $\mathbf{x}_{i-1}$ are the two closest points to $\mathbf{x}_i$ on each side, respectively. Since we deal with a set of Pareto points that are non-dominated, $\mathbf{x}_{i+1}$ and $\mathbf{x}_{i-1}$ are closest to $\mathbf{x}_i$ in terms of both objectives. Both end points of the point sequence are excluded from crowding distance calculation. The overall crowding distance for the objective f_k is

$$\bar{q}^k = \frac{1}{n-2} \sum_{i=2}^{n-1} q_i^k. \tag{29}$$

A smaller crowding distance $\bar{q}^k$ indicates that these solution points are more crowded in terms of the objective f_k and thus the distribution quality is low. Larger crowding distances are desirable since they represent a good distribution where all solution points are more representative.

Optimization techniques are required to explore the objective space as much as possible in order to find more Pareto points. However, among all found points, some can be very close to others. They are not representative and thus should be removed. The question is then which points to remove so that the overall crowding distance can be maximized. First, for each point, we calculate the two crowding distances corresponding to the two objectives. Two ranks r^1 and r^2 are assigned to it based on the comparison in crowding distances with other points. For instance, if the point $\mathbf{x}_i$ has the maximum crowding distance in terms of f_1 among all n points, then $r_i^1 = 1$. If $\mathbf{x}_i$ has the minimum crowding distance, then $r_i^1 = n$. The overall rank of $\mathbf{x}_i$ is

$$r_i^0 = \rho_1 r_i^1 + \rho_2 r_i^2, \tag{30}$$

where ρ_1 and ρ_2 are importance factors of two objectives. These values depend on the application and

$$\rho_1 + \rho_2 = 1. \tag{31}$$

For example, if in an application, f_1 is the single key objective, then we can set ρ_1 to be 1 and ρ_2 to be 0. In this case, r_i^0 is equal to r_i^1 and all points are ranked according to their crowding distances in terms of f_1. After each point $\mathbf{x}_i$ has an overall rank r_i^0, the point with the smallest r_i^0 is removed from the solution set. The entire process starting from crowding distance calculation is iterated until the desirable number of points n_d is reached. Both end points of the point sequence are always kept in the set to maintain the coverage of the solution set. The algorithm is shown in Algorithm 1. It is noted that there are two objectives in this embedded controller design problem for EVs. But everything presented in this section can be easily extended to a multi-objective optimization problem.

Algorithm 1: Removal of less representative solution points according to crowding distance ranking

Input: $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, n_d, $\{\rho_1, \rho_2\}$, $\{f_1, f_2\}$
Output: $S_d = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_d}\}$
for $j \in \{1, 2, \ldots, n - n_d\}$ **do**
  for $i \in \{1, 2, \ldots, n - j + 1\}$ **do**
    calculate q_i^1 and q_i^2 for the element $\mathbf{x}_i$ as in (28);
  end
  for $k \in \{1, 2\}$ **do**
    sort S based on q_i^k from maximum to minimum;
    for $i \in \{1, 2, \ldots, n - j + 1\}$ **do**
      assign the position value (1 for maximum to $n - j + 1$ for minimum) of $\mathbf{x}_i$ in S to r_i^k;
    end
  end
  for $i \in \{1, 2, \ldots, n - j + 1\}$ **do**
    calculate r_i^0 as in (30);
  end
  Sort S based on r_i^0 from maximum to minimum to be $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-j+1}\}$;
  Remove $\mathbf{x}_{n-j+1}$ from S;
end
$S_d = S$;

B. Optimization for Phase II: battery- and aging-aware controller design

As discussed in Section IV, processor aging prolongs the control system sampling period, opening up the possibility of QoC deterioration. This has to be prevented for safety-critical applications like motor control in EVs as it endangers lives of both passengers and the driver. For instance, in adaptive cruise control, if it takes more than original time

Table I
PARAMETERS AND CONSTRAINTS OF THE MOTOR CONTROL SYSTEM

$J\ [kgm^2]$	$b\ [Nms]$	$K_t\ [Nm/A]$	$K_e\ [V/(rad \cdot s)]$	$R\ [\Omega]$	$L\ [H]$	$V\ [V]$	n_s	n_p	$t_s^0\ [s]$	$I_c^{max}\ [A]$	$I_m^{max}\ [A]$
0.15	0.03	0.1	0.1	1	0.01	370	100	30	1.5	5	300

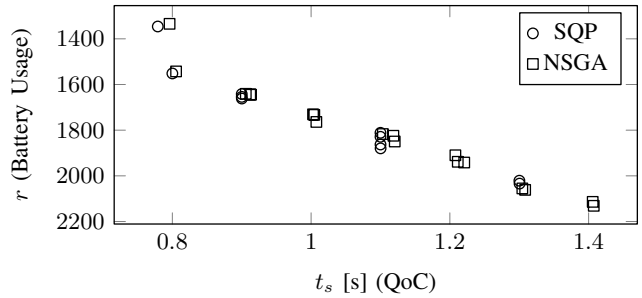

Figure 7. Non-dominated solutions found by SQP and converged solutions found by NSGA. For SQP, 20 random starting points and 3 pairs of weights for objective conversion are used. For NSGA, 20 random starting points are used and the maximum allowed number of generations is 10. Many NSGA solutions are dominated by SQP solutions but NSGA covers more of the objective space.

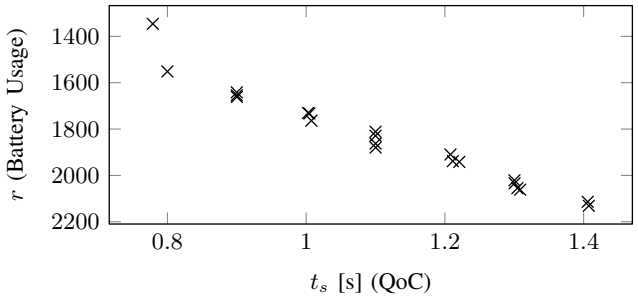

Figure 8. Combined solutions found by both SQP and NSGA. Dominated ones are removed.

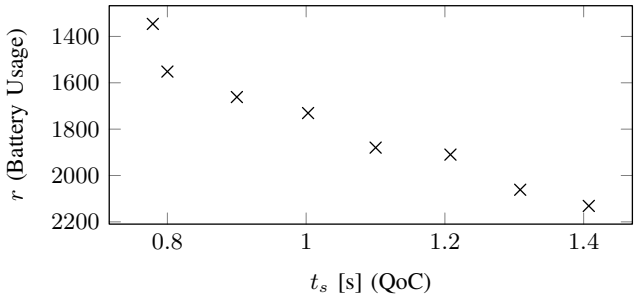

Figure 9. Final non-dominated solutions for Phase I of the embedded control system design. The distribution quality improvement algorithm is applied.

to change the motor speed after the vehicle is driven for some time, accidents might occur. In order to ensure that QoC does not get degraded, i.e., $Q_b' \geq Q_b$, we move to Phase II, which is battery- and aging-aware controller design, as shown in Figure 1. The optimization framework for battery-aware controller design in Phase I is modified. One constraint is added for every solution generated in Phase I that t_s must not get longer. The constraint $t_s \leq t_s^0$ is then not needed any more. Battery usage r becomes the single objective in this battery- and aging-aware controller design optimization problem. The goal will only be approaching the global optimum as closely as possible. In order to achieve it, we need to fully explore the objective space, find local optima and compare them. Therefore, both SQP and the genetic algorithm are used for their respective advantages. For SQP, no objective conversion is necessary. A simpler genetic algorithm is implemented to replace NSGA in the sense that sorting is only based on the objective value r in environmental selection. The algorithm to improve distribution quality is removed as we look for one single solution and the distribution is no longer relevant.

VI. EXPERIMENTAL RESULTS

In this section, we first show the optimization results of the battery-aware embedded control systems design for EVs as illustrated in Phase I of Figure 1. Solutions produced by SQP and NSGA are combined and sorted by domination. Only non-dominated ones are kept. The algorithm is illustrated to improve the distribution quality of the solution set. The trade-off between QoC Q_b and battery usage R_b is explored. Then processor aging effects are reported in terms of both design objectives: QoC Q_b' and battery usage R_b'. It is shown that with controller design re-optimization in battery- and aging-aware controller design of Phase II, the safety-critical QoC can be kept not getting degraded with a slight compromise on battery usage, i.e., $Q_b' \geq Q_b$.

System description: We consider the control task of changing the electric motor speed from $4800\,RPM$ to $7200\,RPM$. The starting state $[\dot{\theta}_0 \quad i_0]^T$ is $[4800\,RPM\ 40\,A]^T$. The output target y_{ref} is $7200\,RPM$. The sampling period h of the system is $0.1\,s$. Parameters of the entire motor control system are shown in Table I. The operating voltage V of the battery pack is $370\,V$, consisting of 100 cells in series and 30 cells in parallel. Each cell has a unit voltage of $3.7\,V$, as presented in Section IV-A. Constraints of the control system design are also shown in Table I. The settling time is expected to be below $1.5\,s$. The maximum allowed current in the battery cell I_c^{max} is $5\,A$ and the maximum allowed current in the motor I_m^{max} is $300\,A$, for the sake of thermal safety.

A. Phase I: battery-aware controller design

Solutions produced by both SQP and NSGA in the objective space are shown in Figure 7. For SQP, three pairs of weights $(w_1 = 1, w_2 = 0)$, $(w_1 = 0, w_2 = 1)$ and $(w_1 = 0.5, w_2 = 0.5)$ are used for objective conversion and 20 starting points are randomly selected. All points converge to local optima and are sorted by domination. Only non-dominated solutions remain. For NSGA, 20 random starting points are used and the maximum allowed number of generations is 10. Converged solutions are automatically non-dominated. From the results, it can be seen that SQP tends to find better points locally but fails to explore certain regions of the objective space. Some of the solutions generated by NSGA are not locally optimal and dominated by solutions from SQP. But NSGA covers more of the

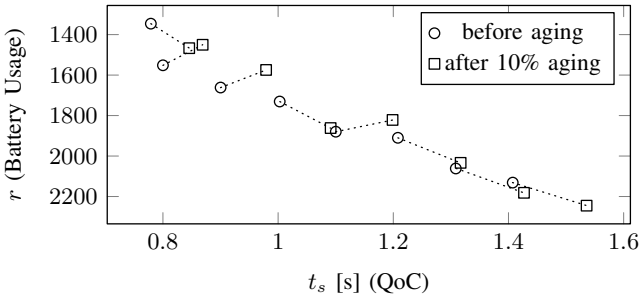

Figure 10. Aging effects in both settling time and battery usage. Two points for the same design case before and after aging are connected with a dotted line. QoC gets deteriorated for all eight solutions generated in Phase I if the controller design does not change. Exact values can be found in Table II.

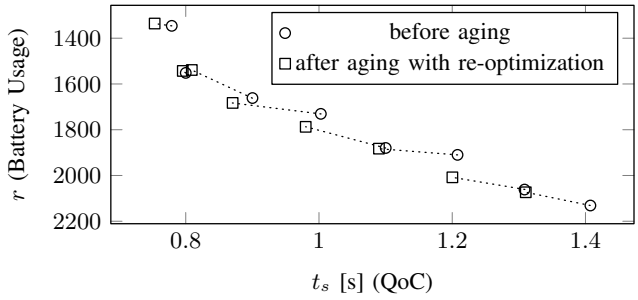

Figure 11. Aging effects mitigation with controller design re-optimization in Phase II. Two points for the same design case before aging and after aging with controller re-optimization are connected with a dotted line. For all eight solutions generated in Phase I, QoC is kept not deteriorated with a slight compromise on battery usage.

objective space and provides more choices to select for a better distribution. These findings match our analysis on both optimization techniques in Section V. All solutions from SQP and NSGA are combined and sorted by domination. Only non-dominated ones are left and presented in Figure 8. Compared with solutions generated by SQP or NSGA alone, combined results are better in objective values and cover more of the objective space.

From the distribution point of view, some of the points in Figure 8 are very close to others and thus redundant. There is still space to improve the distribution quality. Based on the method explained in Section V, we calculate the two crowding distances to be $\bar{q}^1 = 0.065$ and $\bar{q}^2 = 70.925$. Then Algorithm 1 is taken to improve the distribution quality. In this work of embedded controller design for EVs, the objective of QoC $f_1 = t_s$ is safety-critical and more relevant to the vehicle performance. Therefore, we set $\rho_1 = 1$ and $\rho_2 = 0$ as we are more concerned of the distribution quality in f_1. It is noted that other pairs of importance factors for distribution quality can be used as well based on the designer's choice. Assuming that the desirable number of solutions n_d is 8, Algorithm 1 is executed and the final results are shown in Figure 9. It is visually obvious that the distribution quality is improved. Quantitatively, the two crowding distances become $\bar{q}^1 = 0.190$ and $\bar{q}^2 = 215.860$, respectively. Approximately, for every interval of $0.1\,s$ in settling time, there is one solution to choose. The trade-off between QoC Q_b and battery usage R_b can be clearly seen in Figure 9. If we want to pick up a solution point with better QoC, then battery usage has to be compromised, and vice versa. The designer can then make the selection based on different cases.

B. Phase II: battery- and aging-aware controller design

The change in processor operating frequency leads to a decreased sampling frequency of the controller, for which the safety-critical design objective QoC could get degraded. In the following experiments, we assume that the sampling period is increased by $10\,\%$, as discussed in Section IV-B from $0.1\,s$ to $0.11\,s$. In Figure 10 and Table II, we show the aging effects in settling time and battery usage for the eight controller designs generated for Phase I in Figure 9, assuming that the controller eigenvalues do not change. The two points with the same controller design before and after aging are connected with a dotted line. We find that the battery usage and settling time get changed when the processor ages and the control system sampling frequency decreases. Compared to R_b, the battery usage R_b' gets degraded in some cases and improved in other cases. Compared to Q_b, the safety-critical design objective QoC Q_b' gets deteriorated for every solution. In one case, even the hard constraint $t_s \leq 1.5s$ is violated.

As discussed in Section V, it is necessary to ensure that the QoC does not get worse even when the processor ages, i.e., $Q_b' \geq Q_b$. With controller design re-optimization in Phase II, results are shown in Figure 11 and Table II. The two points connected with a dotted line represent the same design case before aging and after aging with controller re-optimization. One can see that it is possible to find new controller design eigenvalues that ensure no degradation of QoC. The price we pay is that the battery usage has slightly deteriorated for all design cases. It is noted that which sampling period is optimal for embedded systems control in EVs is a challenging question to answer motivating future work, since both objectives need to be considered, together with constraints and aging effects. The framework presented in this paper guarantees no compromise in QoC due to processor aging, given any original sampling period. Besides, controller design re-optimization is an offline task. A look-up table with controller eigenvalues corresponding to processor operating frequency is stored in the controller memory. There is a small safety margin to accommodate slight processor aging. When the extend of aging exceeds this safety margin, on-chip monitors realize that it is necessary to reduce the processor frequency. The frequency generator makes the change, and the controller adjusts its eigenvalues accordingly. With a larger look-up table, a smaller safety margin can be set and thus the resource utilization can be improved.

VII. Concluding Remarks

In this paper, we presented a design optimization framework for embedded control systems in EVs — explicitly studying various trade-offs among (i) QoC (ii) battery usage and (iii) processor aging. The results presented in Phase I (battery-aware controller) show that a higher QoC implies a lower battery usage and vice versa. Our framework essentially offers the possibility to achieve trade-offs from the

Table II
AGING EFFECTS IN SETTLING TIME AND BATTERY USAGE WITH AND WITHOUT CONTROLLER DESIGN RE-OPTIMIZATION. DEGRADATION IS CALCULATED WITH RESPECT TO OBJECTIVE VALUES BEFORE AGING. NEGATIVE NUMBERS INDICATE IMPROVEMENT.

Point Number	1	2	3	4	5	6	7	8
t_s before aging [s]	0.779	1.1002	0.9002	0.8	1.0028	1.4074	1.3085	1.208
t_s after aging w/o re-optimization [s]	0.8456	1.1988	0.9792	0.869	1.0912	1.5357	1.427	1.3169
Degradation in t_s w/o re-optimization [%]	8.55	8.96	8.78	8.62	8.82	9.11	9.05	9.02
t_s after aging with re-optimization [s]	0.7534	0.9801	0.8091	0.7956	0.8704	1.3104	1.2002	1.0900
Degradation in t_s with re-optimization [%]	-3.29	-10.92	-10.11	-0.56	-13.2	-6.89	-8.28	-9.77
r before aging	1346	1880	1662	1552	1731	2131	2062	1910
r after aging w/o re-optimization	1467	1822	1575	1450	1862	2245	2182	2034
Degradation in r w/o re-optimization [%]	-8.99	3.09	5.23	6.54	-7.57	-5.34	-5.83	-6.48
r after aging with re-optimization	1336	1787	1539	1543	1683	2073	2008	1883
Degradation in r with re-optimization [%]	0.75	4.92	7.38	0.55	2.76	2.72	2.59	1.42

Pareto front. Further, when the processor ages, QoC gets deteriorated. We utilized a modified version of our optimization framework in Phase II (battery- and aging-aware controller) and experimented considering $10\,\%$ reduction in processing speed due to aging. Our results show that QoC deterioration due to aging can be mitigated by a little compromise on battery usage. Thus, our framework allows to preserve the safety-critical guarantees in spite of processor aging at the cost of a slightly lower battery usage.

ACKNOWLEDGMENT

This work is supported by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) program, the project ARTEMIS-2013-1 621429 EMC2, and the Bavarian Ministry of Economic Affairs and Media, Energy and Technology as part of the EEBatt project.

REFERENCES

[1] Y. Hyunbean, T. Yoneda, I. Inoue, Y. Sato, S. Kajihara and H. Fujiwara, *A failure prediction strategy for transistor aging*, IEEE Transactions on Very Large Scale Integration Systems, vol. 20, no. 11, pp. 1951-1959, 2012.

[2] D. Doerffel and S. Sharkh, *A critical review of using the Peukert equation for determining the remaining capacity of lead-acid and lithium-ion batteries*, Journal of Power Sources, vol. 165, no. 2, pp. 395-400, 2006.

[3] T. Kim and W. Qiao, *A hybrid battery model capable of capturing dynamic circuit characteristics and nonlinear capacity effects*, IEEE Transactions on Energy Conversion, vol. 26, no. 4, pp. 1172-1180, 2011.

[4] D. Lorenz, M. Barke and U. Schlichtmann, *Aging analysis at gate and macro cell level*, ICCAD, pp. 77-84, 2010.

[5] M. Haddad and V. Kapila, *A periodic fixed-structure approach to multirate control*, CDC, pp. 1791-1796, 1993.

[6] M. Mizuochi, T. Tsuji and K. Ohnishi, *Multirate sampling method for acceleration control system*, IEEE Transactions on Industrial Electronics, vol. 54, no. 3, pp. 1462-1471, 2007.

[7] E. Bini and A. Cervin, *Delay-aware period assignment in control systems*, RTSS, pp. 291-300, 2008.

[8] D. Henriksson and A. Cervin, *Optimal on-line sampling period assignment for real-time control tasks based on plant state information*, CDC, pp. 4469-4474, 2005.

[9] R. Castane, P. Marti, M. Velasco, A. Cervin and D. Henriksson, *Resource management for control tasks based on the transient dynamics of closed-loop systems*, ECRTS, pp. 171-182, 2006.

[10] F. Zhang, K. Szwaykowska, W. Wolf and V. Mooney, *Task scheduling for control oriented requirements for cyber-physical systems*, RTSS, pp. 47-56, 2008.

[11] K. Smith, C. Rahn and C. Wang, *Control oriented 1D electrochemical model of lithium ion battery*, Energy Conversion and management, vol. 48, no. 9, pp. 2565-2578, 2007.

[12] S. Hageman, *Simple PSpice models let you simulate common battery types*, Electronic Design News, vol. 38, pp. 117-129, 1993.

[13] D. Rakhmatov and S. Vrudhula, *An analytical high-level battery model for use in energy management of portable electronic systems*, ICCAD, pp. 488-493, 2001.

[14] A. Masrur, P. Kindt, M. Becker, S. Chakraborty, V. Kleeberger, M. Barke and U. Schlichtmann, *Schedulability analysis for processors with aging-aware autonomic frequency scaling*, RTCSA, pp. 11-20, 2012.

[15] C. Lefurgy, A. Drake, M. Floyd, M. Alle, B. Brock, J. Tierno and J. Carter, *Active management of timing guardband to save energy in POWER7*, MICRO, pp. 1-11, 2011.

[16] K. Bowman, J. Tschanz, C. Wilkerson, S. Lu, T. Karnik, V. De and S. Borkar, *Circuit techniques for dynamic variation tolerance*, DAC, pp. 4-7, 2009.

[17] K. Astrom and R. Murray, *Feedback systems: an introduction for scientists and engineers*, Princeton University Press, 2009.

[18] O. Ljungkrantz, H. Lonn, H. Blom, C. Ekelin and D. Karlsson, *Modelling of safety-related timing constraints for automotive embedded systems*, SAFECOMP, vol. 7613, pp. 190-201, 2012.

[19] M. Ruderman, J. Krettek, F. Hoffmann and T. Bertram, *Optimal state space control of DC motor*, IFAC, pp. 5796-5801, 2008.

[20] Sony, *Lithium ion rechargeable batteries technical handbook [online]*, www.sony.com.cn/products/ed/battery/download.pdf.

[21] R. Mishra, D. Ioannou, S. Mitra and R. Gauthier, *Effect of floating-body and stress bias on NBTI and HCI on 65-nm SOI pMOSFETs*, IEEE Electron Device Letters, vol. 29, no. 3, pp. 262-264, 2008.

[22] D. Schroder and J. Babcock, *Negative bias temperature instability: road to cross in deep submicron silicon semiconductor manufacturing*, Journal of Applied Physics, vol. 94, no. 1, pp. 1-18, 2003.

[23] J. Park and J. Abraham, *A fast, accurate and simple critical path monitor for improving energy-delay product in DVS systems*, ISLPED, pp. 391-396, 2012.

[24] A. Kumar, D. Sharma and K. Deb, *A hybrid multi-objective optimization procedure using PCX based NSGA-II and sequential quadratic programming*, CEC, pp. 3011-3018, 2007.

[25] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, 2002.