Quality Estimation and Optimization of Adaptive Stereo Matching Algorithms for Smart Vehicles

Fupeng Chen, Heng Yu, and Yajun Ha

University of Nottingham
UK | CHINA | MALAYSIA

Faculty of Science and Engineering, University of Nottingham Ningbo China, 199 Taikang East Road, Ningbo, 315100, Zhejiang, China.

First published 2020

University of Nottingham
UK | CHINA | MALAYSIA

Quality Estimation and Optimization of Adaptive Stereo Matching Algorithms for Smart Vehicles

FUPENG CHEN*†, School of Information Science and Technology, ShanghaiTech University, China
HENG YU, University of Nottingham Ningbo China, China
YAJUN HA, School of Information Science and Technology, ShanghaiTech University, China

Stereo matching is a promising approach for smart vehicles to find the depth of nearby objects. Transforming a traditional stereo matching algorithm to its adaptive version has potential advantages to achieve the maximum quality (depth accuracy) in a best effort manner. However, it is very challenging to support this adaptive feature, since (1) the internal mechanism of adaptive stereo matching (ASM) has to be accurately modeled, and (2) scheduling ASM tasks on multiprocessors to generate the maximum quality is difficult, under strict real-time constraints of smart vehicles. In this paper, we propose a framework for constructing an ASM application and optimizing its output quality on smart vehicles. First, we empirically convert stereo matching into ASM by exploiting its inherent characteristics of disparity-cycle correspondence and introduce an exponential quality model that accurately represents the quality-cycle relationship. Second, with the explicit quality model, we propose an efficient quadratic programming-based dynamic voltage/frequency scaling (DVFS) algorithm to decide the optimal operating strategy, which maximizes the output quality under timing, energy, and temperature constraints. Third, we propose two novel methods to efficiently estimate the parameters of the quality model, namely location similarity-based feature point thresholding (L-FPT) and street scenario-confined CNN (S-CNN) prediction. Results show that our DVFS algorithm achieves at least 1.61 times quality improvement compared to the state-of-the-art techniques, and average parameter estimation for the quality model achieves 96.35% accuracy on the straight road.

CCS Concepts: • **Computing methodologies** → *Matching*; *Neural networks*; • **Computer systems organization** → *Embedded software*; • **Hardware** → *Operations scheduling*.

Additional Key Words and Phrases: Binocular Stereo Matching, Smart Vehicle, Adaptive Application, Embedded Systems

ACM Reference Format:
Fupeng Chen, Heng Yu, and Yajun Ha. 2020. Quality Estimation and Optimization of Adaptive Stereo Matching Algorithms for Smart Vehicles. 1, 1 (March 2020), 25 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*Also with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences.
†Also with University of Chinese Academy of Sciences.

Authors' addresses: Fupeng Chen, School of Information Science and Technology, ShanghaiTech University, 393 Middle Huaxia Road, Pudong, Shanghai, China, 201210, chenfp@shanghaitech.edu.cn; Heng Yu, University of Nottingham Ningbo China, 199 Taikang East Road, Ningbo, China, 315100, heng.yu@nottingham.edu.cn; Yajun Ha, School of Information Science and Technology, ShanghaiTech University, 393 Middle Huaxia Road, Pudong, Shanghai, China, 201210, hayj@shanghaitech.edu.cn.

XXXX-XXXX/2020/3-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Background

Stereo matching algorithms are key enablers for the perceptibility of smart vehicles, empowering various imperative functionalities such as traffic objects detection and tracking [6, 8, 30, 37], visual odometry [13], and 3D reconstruction [36]. Compared to active sensory systems such as those employing LIDAR [38], stereo matching-based solutions are typically cheaper and more versatile.

Adaptive stereo matching (ASM) algorithms, unlike normal ones, are able to provide scalable execution quality in reaction to the execution environment. The more execution cycles are assigned to ASM algorithms, the higher quality it can achieve in the output, at the price of increased system timing/energy resources [40]. For example, when more cycles are assigned, the larger disparity range is then allowed and increasingly accurate matching (quality) results can be obtained.

Accurately modeling the relationship between quality and execution cycles is challenging for ASM algorithms. Traditionally, quality modeling is implemented by extensive profiling and curve fitting [41]. We have seen past researchers having used linear [25, 40, 45] or exponential quality models [41]. It needs substantial work to decide which model suits an algorithm the most. In addition, each model may have its parameters, and the values of these parameters are usually input-dependent. For example, even after we have chosen to use an exponential model, we still need to further decide the values for several parameters in the exponential model. As a result, it is essential to decide not only a suitable quality model for ASM algorithms but also its best parameter values.

With simplified quality models, some DVFS-based optimization approaches have been developed to maximize the total cycles under timing and energy constraints [25, 40, 45]. A common assumption taken by those works is that maximizing quality can be implicitly achieved by maximizing the processor execution cycles of tasks. While the assumption makes the problem formulations and solutions easier, ignoring the non-linear quality-cycle relationship could produce suboptimal quality maximization results. Instead of being cycle-centric, we would like to develop a quality-centric DVFS formulation and solution in this work.

In total, several important issues have to be addressed before this promising ASM can be optimally and practically employed on smart vehicular systems, including how to: (1) Accurately model the output quality adaptivity versus execution cycles; (2) Optimally determine the system execution parameters, such as processor operating voltage/frequency and execution cycles, that achieve the quality maximization; and (3) Accurately estimate the parameters of the quality-cycle model. Since the parameters are input image-dependent, extensive profiling of the stereo matching may still lead to inaccuracy.

1.2 Illustrative Example

Being adaptive may not automatically guarantee optimized output quality for ASM tasks. In this part, we demonstrate the effectiveness of judiciously configuring system V/F settings (as well as pre-determining ASM cycle values) for ASM tasks to maximize the total execution quality, being aware of their quality models. Assume two stereo matching tasks, T_1 from Kitti datasets [9] and T_2 from Carla datasets [7], whose execution cycles/frequencies/voltages are to be determined before actually running. The tasks feature attenuatively increasing quality functions over each cycle, and we adopt the corresponding exponential function to model them, as described in Table 1. We also assume that the task's energy consumption is linearly related to execution cycles, under a given voltage v_{dd}. It is expressed as $E = Kv_{dd}^2 \times o$, where K is the chip-specific constant and o is the number of execution cycles. The initial conditions of each task can be found in Table 1 and Fig. 1(a). The execution cycle o can be calculated as the product of executing time and frequency. The total

quality of execution T_1 and T_2 is calculated as $7.3(1-e^{-\frac{0.9e+8}{0.04e+9}})+6.7(1-e^{-\frac{1.8e+8}{0.03e+9}})=13.21$. The matching error Err, to be defined later as the inverse of the quality, of T_1 and T_2 are $Err1=15.3\%$ and $Err2=14.96\%$, respectively.

Table 1. Properties of T_1 and T_2 for the stereo matching example.

Task	Exe. Time	Init. Freq	Init. V_{dd}	Deadline	Qual. Function
T_1	300ms	300MHz	0.9v	600ms	$7.3(1-e^{-\frac{o}{0.04e+9}})$
T_2	600ms	300MHz	0.9v	600ms	$6.7(1-e^{-\frac{o}{0.03e+9}})$

Fig. 1(a) shows the effects of different cycle allocation methodologies to achieve quality maximization. Assume that instead of executing for 600ms, T_2 finishes at 400ms, and the spared processor cycles, which is equal to 300MHz×200ms = 6E+7 cycles depicted in area A in Fig. 1, are allocated to T_1 running at 300MHz. The new total quality then becomes $7.3(1-e^{-\frac{1.5e+8}{0.04e+9}})+6.7(1-e^{-\frac{1.2e+8}{0.03e+9}})=13.71$. The matching error, Err, of T_1 and T_2 are $Err1=14\%$ and $Err2=15.2\%$, respectively. The quality improvement is subject to the awareness of quality function attenuation: Comparing the two quality functions, T_2 executing from 400ms to 600ms does not increase as much quality as T_1 does from 300ms to 500ms. This shows that the total execution quality can be optimized if system resources are located being aware of the quality function characteristics, especially quality attenuation. Also, fast and accurate quality function estimation is an imperative prerequisite for quality maximization.

Fig. 1(b) illustrates how judiciously applying DVFS could help further improve the execution quality. Here the T_2's voltage/frequency is reduced from {0.9v, 300MHz} to {0.8v, 200MHz}. We accordingly extend the execution cycles of T_2 to 600ms to keep the cycle and quality of T_2 unchanged. The reduction of T_2's voltage/frequency saves energy of $\triangle E=E_{2,300MHz}-E_{2,200MHz}=K\cdot(0.81-0.64)\cdot 1.2\times 10^8$ units. If $\triangle E$ is claimed by T_1, then additional cycles and quality can be generated, as depicted in area B in Fig. 1(b). Assume that T_1 still runs at {0.9v, 300MHz}, the new execution cycles become 1.75E+8 after claiming $\triangle E$. The total quality of T_1 then becomes $7.3(1-e^{-\frac{1.75e+8}{0.04e+9}})+6.7(1-e^{-\frac{1.2e+8}{0.03e+9}})=13.79>13.71>13.21$, indicating that the DVFS further improves the execution quality of adaptive tasks. The corresponding matching error, Err, of T_1 and T_2 are $Err_1=13.87\%$ and $Err_2=15.2\%$, respectively. Compared to the initial condition, T_1's error is reduced by 9.35% at the cost of increasing T_2's error by 1.6%.

The visual illustration of the effects of applying the abovementioned optimizations is shown in Fig. 2. With judicious cycle allocation and DVFS strategies, it is possible to substantially improve the overall stereo matching quality under identical resource constraints. In this work, we propose a framework that efficiently estimates the quality function for the stereo matching application, takes both quality attenuation and DVFS into the problem formulation, and provides an efficient solution to find the optimal execution settings of cycle, voltage, and frequency.

1.3 Related Work

Adaptive applications have been studied extensively given its flexibility such that the system resources can be adjusted in response to changes in the environment. In [23], the authors summarize current methods and applications of adaptive systems. There have been pioneering works modeling quality-scalable applications, including Imprecise-Computation tasks [5] and multi-version tasks [33]. Aydin *et al.* provide an optimal static solution for the imprecise computation task scheduling problem using convex programming [2]. An approximate computing framework, named

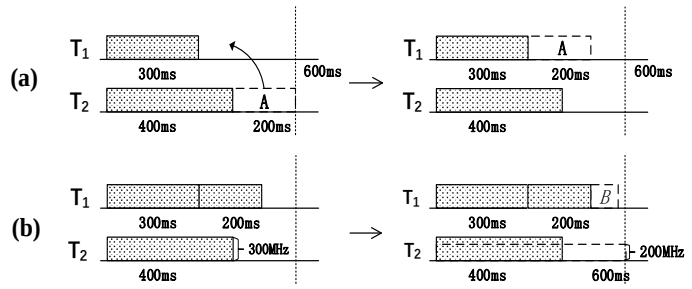

Fig. 1. Stereo matching example: (a) Considering quality function characteristics helps optimally allocate the system resources to obtain higher quality. (b) Applying DVFS further improves output quality. Initial settings of (b) continue from (a).

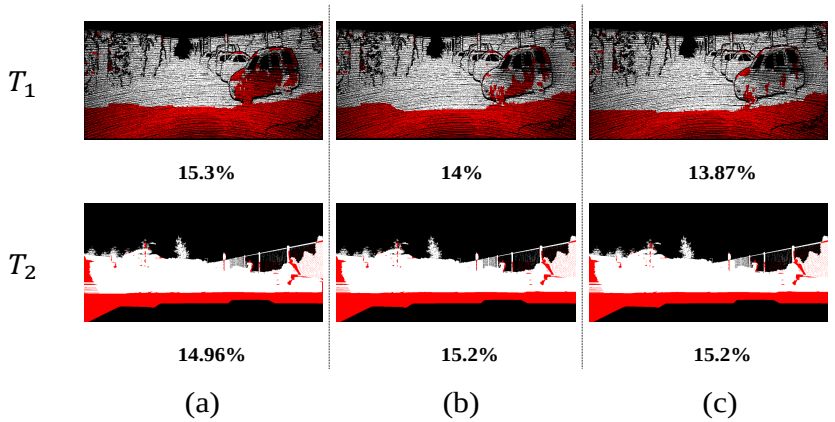

Fig. 2. Illustration of the ASM matching error, *Err*, (indicated in the red region) after cycle re-allocation and DVFS optimization.The top row shows the results executing task T_1, whose input images are obtained from Kitti; the bottom row shows executing task T_2 whose input is from Carla. (a) illustrates an initial matching error; (b) shows the results after for judicious cycle re-allocation; (c) shows the results after applying DVFS for further optimization.

ApproxIT, is proposed to manage the quality-scalable applications dynamically to guarantee the output quality, by unfolding the execution of iterative methods [43]. Chippa *et al.* propose the concept of dynamic effort scaling, which guides the feedback control to scale the system execution quality dynamically at different levels of abstractions [4].

There exist research works demonstrating that stereo matching methods are inherently suitable to be converted into adaptive versions [18, 21, 22, 39]. Adaptive window methods [21, 22] try to find an optimal support window to determine the size of matching and improve the matching accuracy. An adaptive guided image filter [39] is proposed to feature an adaptive rectangular support window instead of the traditionally fixed window. The adaptive support weight method [18] attempts to adjust the support-weights of the pixels in a given window to reduce the image ambiguity. Unfortunately, none of the works models the workload variations that the adaptivity could bring to processors.

On the other hand, several representative stereo matching algorithms are potentially suitable to be converted into the adaptive version with respect to processor cycles. The Efficient LArge-scale Stereo (ELAS) [10] is based on a generative probabilistic model, and the authors propose a Bayesian-based approach to compute accurate disparity maps. It builds a prior distribution over the disparity space by forming a triangulation on robust support points, decreasing stereo matching ambiguities without the need for global optimization. The bilateral filter [28] can decompose an image into different scales without causing haloes after modification while respecting strong edges. The guided filter can generate the filtering output by considering the content of a guidance image, derived from a local linear model [12]. It has the edge-preserving property and is runtime independent of the filter size. An undeveloped feature of these algorithms is that they can change their execution time by adjusting the disparity parameter to convert them into adaptive versions. In this paper, we investigate the internal mechanisms of these stereo matching algorithms to exploit the possibility of turning one into its adaptive version.

Dynamic Voltage and Frequency Scaling (DVFS) allows processors to change the voltage and corresponding frequency dynamically, which in turn alter the behaviors of the workload execution, which affects energy consumption, makespan, heat generation, etc. Traditionally, DVFS-based methodologies optimize the performance metrics such as energy [27] and throughput [11, 14, 44]. The problems are usually converted into optimization formulations with one or more of those metrics as objectives, and remaining ones as constraints that trade off the objectives [2, 26, 40]. Generally, an important assumption of the existing works is that the task execution cycles are non-adaptive, making them less applicable when applied to adaptive workloads.

Recently, DVFS-based optimization approaches have been observed on quality-adaptive applications, to maximize total quality under timing, energy, and thermal constraints [25, 40–42, 45]. Mo *et al.* proposed to maximize the optional cycles of imprecise computation tasks by proposing formulations and solutions based on mixed-integer linear programming [25]. Zhou *et al.* presented adaptive task mapping and scheduling heuristics targeting quality maximization under renewable energy supplies [45]. A scheduling strategy targeting adaptive task dependency and communication is proposed in [40]. While the above works strive to achieve the optimal execution cycles, they ignore the non-linear quality-cycle mappings that may lead to a sub-optimal decision. In this work, we consider the application-specific quality-cycle properties and directly optimize the quality. Authors in [41] propose an efficient iterative pseudo quadratic programming heuristic to decide the optimal cycle using convex quality functions. However, the approach considers only frequency scaling under a given voltage, while the quality improvement space can be more significant if comprehensively applying DVFS. While all the abovementioned algorithms attempt to distribute the energy budget to individual parallel tasks optimally, naive heuristics (such as Evenly Distributing (ED) the energy budget to parallel tasks) exhibit low algorithmic complexity at the expense of degraded quality output. The efficient formulation and solution of the resource budget distribution are essential to obtain optimal quality at comparable complexity.

1.4 Scope of the paper

In this work, we make the following contributions towards optimizing the total execution quality of adaptive stereo matching applications for smart vehicle systems:

- We exploit the mechanism of converting a representative class of stereo matching applications (namely, the *binocular stereo matching*), into the adaptive version (namely, ASM) that can flexibly adjust the output quality.

- We develop a DVFS-based approach to maximize the output quality of ASM tasks under the system timing, energy, and temperature constraints. Compared to state-of-the-art works, our approach achieves significant quality improvement by:
  - Directly optimizing the system quality metric instead of implicitly maximizing CPU cycles,
  - Judiciously scaling the supply voltage to exploit larger optimization space compared to frequency scaling-based approaches, and
  - Proposing an efficient quadratic programming formulation to achieve quality optimization at low algorithmic overhead.
- We propose two efficient and accurate solutions, namely location similarity-based feature point thresholding (L-FPT) and street scenario-confined CNN (S-CNN), that infer the quality function parameters. We verify the proposed methods using the dataset captured from the Carla simulator [7].

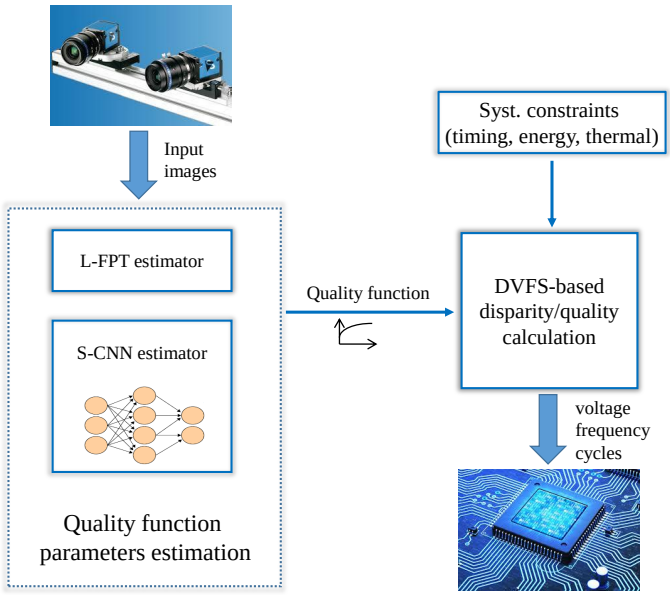

Fig. 3. The framework of the proposed scheme for ASM quality estimation and optimization.

Our work leads to a framework for modeling, executing, and optimizing ASM on smart vehicle platforms, as summarized in Fig. 3. In the following sections, we discuss the components inside the framework in detail. Section 2 introduces the modeling of ASM and timing/energy/temperature constraints. Section 3 presents the DVFS algorithm to maximize output quality. How to estimate quality function parameters using L-FPT and S-CNN is described in Section 4. Section 5 presents the experimental results and analysis. Section 6 concludes the paper.

2 MODELING

2.1 Adaptive Binocular Stereo Matching

Given binocular imagery of the same scene, stereo matching methods attempt to match pixels in one image with the corresponding ones in the other, to enable calculating the object distance. Fig. 4 illustrates a simplified binocular stereo matching system, which captures the scene of an object as a pair of images taken into the paired cameras. The term **disparity** denotes the distance between

two corresponding pixels in the left and right images. The **scene depth**, namely the detectable distance between the object and cameras, is inversely proportional to the disparity value.

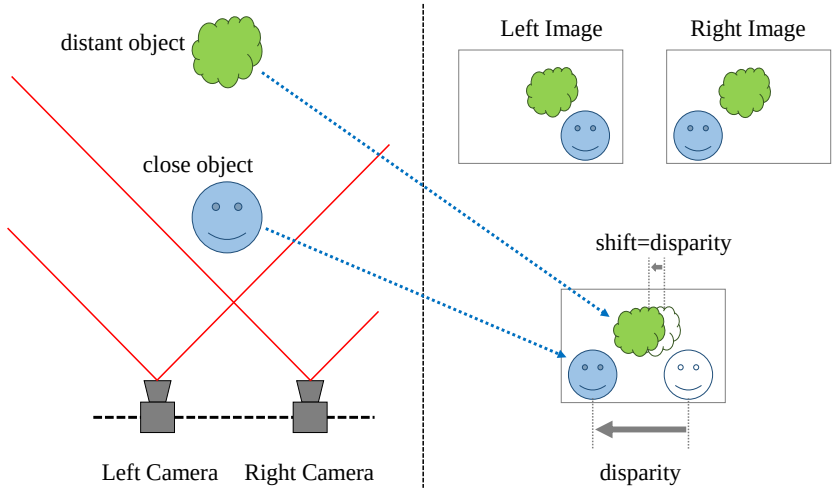

Fig. 4. The simplified binocular matching system. The image pair is rectified. Note that larger disparity implies shorter scene depth.

Many stereo matching algorithms constrain the search space of disparity within a range [d_{min}, d_{max}] to alleviate the computational cost [9, 10, 34]. However, lowering d_{max} brings the side effect of increased matching error rate, since reducing the disparity leads to more regions that are undetectable directly while the paired pixels are located beyond d_{max}, typically the regions near the cameras. Fig. 5 illustrates the matching results of three different images after applying ELAS, where the red region indicates the parts that are not successfully detected and matched.

Definition 1: The matching error of the stereo matching algorithms, denoted as Err, is defined as the percentage of the pixels whose disparity differs from the ground truth (GT) by a certain threshold, according to [35]:

$$Err = \frac{\#pixels\ with(|disp - GT| < threshold)}{\#all\ pixels} \times 100\% \quad (1)$$

On the other hand, reducing the disparity range does reduce the computation cost of the stereo matching algorithms, since the search space for matching shrinks in a smaller disparity range. Fig. 6 shows the profiling results of extensively running several representative stereo matching algorithms, which all show inherent characteristics that the disparity range exhibits a linear relationship with the application execution cycles. Given the linear and inversely proportional relationships among <execution cycle, disparity> and <disparity, error> pairs, respectively, it implies that by carefully manipulating the d_{max} (assume that d_{min} is fixed), the application execution cycle and matching error exhibit a reverse relationship. Simply put, more execution cycles lead to less error (i.e., more quality), and fewer execution cycles give more error (i.e., less quality).

Definition 2: The output quality of stereo matching is defined as $\mathcal{F}(o) = 1/Err$, where o represents the stereo matching execution cycles.

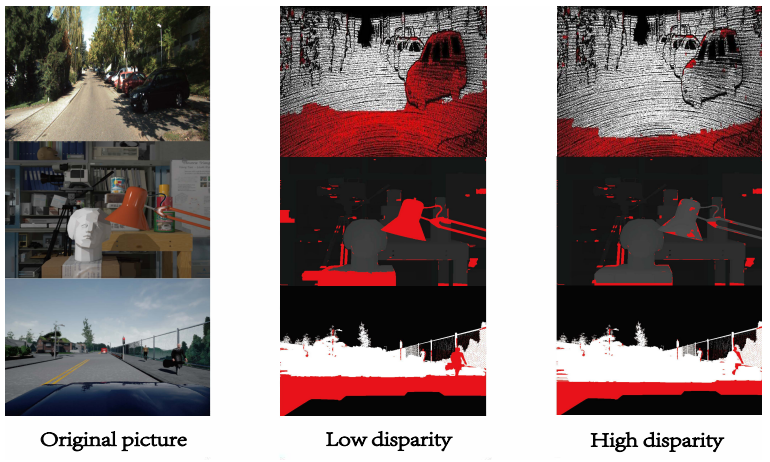

Fig. 5. The matching results of three images after applying ELAS. Red regions are not detected. Left: Original images from three datasets (Kitti [9], New Tsukuba [24], Carla [7]). Middle: Low disparity range. Right: High disparity range.

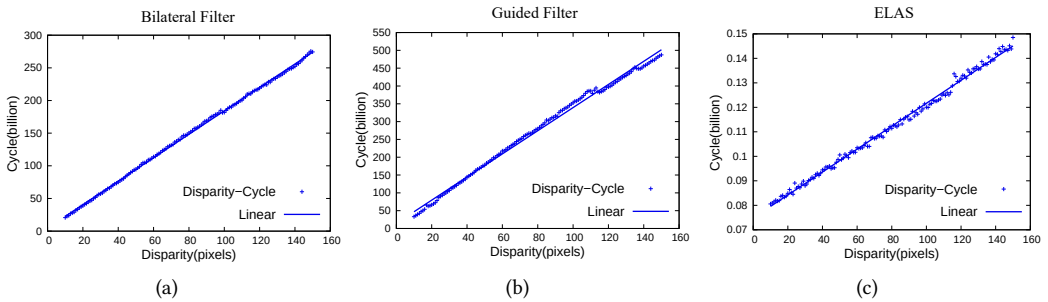

Fig. 6. Linear relationship between disparity range and average execution cycles. Data averaged for 1000 times with different disparity range. Left is bilateral filter, middle is guided filter, and right is ELAS.

To capture the quality-cycle relationship, we examine three models, namely linear, power, and exponential models. Fig. 7 gives an intuitive comparison using the three models. The green dots represent the ASM quality, measured by the inverse of the matching error. The modeling error is evaluated using the Mean Square Error (MSE) between the original quality values and the fitted values. The MSE values of <linear, power, exponential> modeling of the three sample images are <0.3014, 0.0104, 0.0037>, <0.4598, 0.0185, 0.0042>, and <0.2289, 0.1027, 0.0485>, respectively. It shows that the exponential model gives better modeling accuracy than linear and power models.

As quantitatively evaluated in Section 5, exponential functions most accurately model the stereo matching quality-cycle characteristics compared to the other two.[1] In this work, we thus focus on

[1]There may exist other closed-form models, such as polynomial ones, that more accurately describe the quality-cycle characteristics. However, we adopt the straightforward exponential model that favors the optimization formulation and solution proposed in this work. More models that are complicated are subject to future studies.

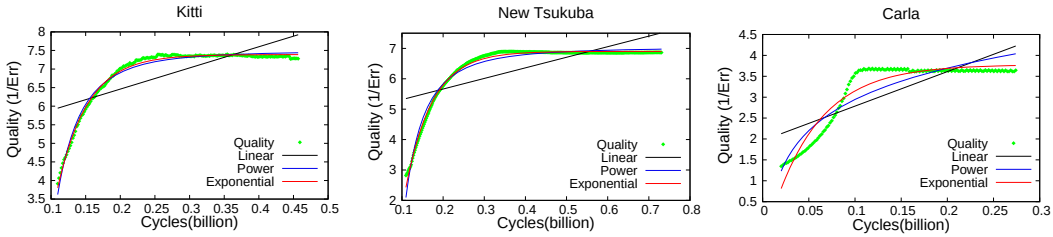

Fig. 7. ASM quality (inverse of error) changes with execution cycles for three sample images from the three datasets (left to right - Kitti, New Tsukuba, Carla).

the following concave quality function:

$$\mathcal{F}_e(o) = a \times \left(1 - e^{-\frac{o}{b}}\right) + m \tag{2}$$

where a, b, m are stereo matching specific function parameters, and o represents the execution cycles. Ideally, the parameters a, b, m are input-invariant and can be obtained through extensive profiling and curve fitting [41]. However, for stereo matching applications, the parameters a, b, m are largely input image dependent. In Section 4, L-FPT and S-CNN solutions are proposed to obtain the parameters efficiently and accurately in real-time.

2.2 Platform

2.2.1 System and Energy Models: We assume that the ASM tasks run on an MPSoC embedded in the vehicular system, including a set of heterogeneous processors $p \in P$ with voltage/frequency scaling capabilities. The power consumption contains both dynamic and static power components. The dynamic power consumption for task i is expressed as:

$$P_{i,dyn} = SW \cdot f_i \cdot v_i^2 \tag{3}$$

where SW is the average switching capacitance, f_i is the processor frequency, and v_i is the processor supply voltage. To emphasize the role that voltage plays, we change the expression to $P_{i,dyn} = C^{f_i} v_i^2$, for processor p running at f_i and voltage v_i. Leakage power can be approximated as a consecutive piecewise-linear model w.r.t the temperature of the thermal block and the voltage of a core [11]. The corresponding expression is given as:

$$P_{i,lkg} = P_{i0,lkg} + K_\alpha T_i + K_\beta v_i \tag{4}$$

where $P_{i0,lkg}$ is the initial power at the given linear section, T_i is the chip temperature. K_α and K_β are the slopes of leakage power for the temperature and the voltage in the given linear section, and can be obtained by extensive profiling [11]. We have then the total power consumption:

$$P_i = C^{f_i} v_i^2 + P_{i0,lkg} + K_\alpha T_i + K_\beta v_i \tag{5}$$

The total energy consumption to execute o_i cycles for task i, which spends time $t_i = \frac{o_i}{f_i}$, is expressed as:

$$E_i = o_i f_i^{-1} (C^{f_i} v_i^2 + P_{i0,lkg} + K_\alpha T_i + K_\beta v_i) \tag{6}$$

In this work, we assume that for a given voltage v_i, the f_i chosen belongs to a corresponding discrete set F_{v_i}, namely $f_i \in F_{v_i}$.

2.2.2 Thermal-Constrained Voltage: The heat dissipation for on-chip thermal blocks can be expressed analogously to a thermal RC model [15]:

$$C_p \frac{dT_i(t)}{dt} + \sum_{m \in M} G_{p,m}(T_i(t) - T_m(t)) + G_{p,env}(T_i(t) - T_{env}) = P_i \tag{7}$$

where $\mathcal{M}$ is the set of all adjacent processors, $G_{p,m}$ is the thermal conductance between p and neighbor processor m, T_{env} is the surrounding air temperature, and $G_{p,env}$ is the thermal conductance to air surroundings, including both chip cover and bottom surfaces. By substituting (5) into (7), and assuming that considering only steady-state temperature, we obtain the relationships of temperature and voltage as the following matrix form:

$$\vec{\mathrm{T}}^l \leq \Psi_1^f \vec{\mathrm{v}}^2 + \Psi_2^f \vec{\mathrm{v}} + \Phi^{f,T} \leq \vec{\mathrm{T}}^u \tag{8}$$

where $\vec{\mathrm{v}} = [v_1, v_2, ..., v_{numP}]^\intercal$, $\vec{\mathrm{T}}^l$ and $\vec{\mathrm{T}}^u$ are the vectored thermal lower and upper limits, Ψ_1^f, Ψ_2^f, and $\Phi^{f,T}$ are coefficient matrices determined by frequency, power, and thermal conductance. The detailed derivation of (8) can be found in [41], where we replace the frequency variable with the voltage under the assumption that $G_{p,m} = 0$ if processor p is not directly adjacent to other processor block m, and $C\frac{dT_i(t)}{dt} \to 0$, namely the temperature quickly converges to steady state.

3 DVFS FOR QUALITY MAXIMIZATION

Given the abovementioned models, in this section, we propose the DVFS formulation and solution to obtain the maximum output quality of ASM tasks under timing, energy, and thermal constraints. The QP formulation is described in Section 3.1. The voltage scaling heuristic is presented in Section 3.2. Section 3.3 presents the complete DVFS algorithm that iteratively solves the QP problem, employing voltage scaling between the QP iterations.

3.1 QP Formulation for Quality Optimization

We focus on maximizing the total quality improvement of all the ASM tasks that run in parallel after being dispatched in the ready queue in a multiprocessor system. The total quality improvement can be expressed as:

$$\sum_i (\mathcal{F}_{e,i}(o_i + \triangle o_i) - \mathcal{F}_{e,i}(o_i)) = \sum_i a_i(-e^{-\frac{o_i + \triangle o_i}{b_i}} + e^{-\frac{o_i}{b_i}}) \tag{9}$$

where a_i, b_i, o_i are task-specific exponential parameters and initial cycles, and Δo_i is the number of cycles to be improved. To formulate the problem into a tractable form, namely the QP form, we apply Taylor's expansion to change the objective expression (9) to be quadratic.

$$\begin{aligned}
\mathcal{F}_{e,i}(o_i+\triangle o_i)/a_i &= \mathcal{F}_{e,i}(o_i) + \mathcal{F}'_{e,i}(o_i)\triangle o_i + \frac{\mathcal{F}''_{e,i}(o_i)}{2}\triangle o_i^2 \\
&\quad + \frac{\mathcal{F}^{(3)}_{e,i}(o_i)}{6}\triangle o_i^3 + \cdots \\
&= (1 - e^{-\frac{o_i}{b_i}}) + \frac{\triangle o_i}{b_i}(e^{-\frac{o_i}{b_i}}) + \frac{\triangle o_i^2}{b_i^2}(-\frac{1}{2}e^{-\frac{o_i}{b_i}}) \\
&\quad + \frac{\triangle o_i^3}{b_i^3}(\frac{1}{6}e^{-\frac{o_i}{b_i}}) + \cdots \\
&\approx \triangle o_i^2(-\frac{1}{2b_i^2}e^{-\frac{o_i}{b_i}}) + \triangle o_i(\frac{1}{b_i}e^{-\frac{o_i}{b_i}}) + (1 - e^{-\frac{o_i}{b_i}})
\end{aligned}$$

Because b_i is constant through modeling exponential function, we can assume $\triangle o_i \leq b_i$, and then higher order (≤ 3) can be ignored due to tolerable errors in our stereo matching adaptability. The objective expression (9) can be approximated as (10) in the following QP formulation:
Minimize

$$\sum_i a_i((\frac{1}{2b_i^2}e^{-\frac{o_i}{b_i}})\triangle o_i^2 + (-\frac{1}{b_i}e^{-\frac{o_i}{b_i}})\triangle o_i) \tag{10}$$

Subject to

$$\sum_i (o_i+\triangle o_i)(C^{f_i}v_i^2+K_\beta v_i+P_{i0,lkg}+K_\alpha T_i)f_i^{-1} \leq \epsilon_s \tag{11}$$

$$\frac{(o_i+\triangle o_i)}{f_i} \leq \tau_s \tag{12}$$

$$\vec{\mathrm{T}}^l \leq \Psi_1^f \vec{\mathrm{v}}^2 + \Psi_2^f \vec{\mathrm{v}} + \Phi^{f,T} \leq \vec{\mathrm{T}}^u \tag{13}$$

The constraints are described as follows:

- **Energy Constraint** (11) is obtained from (6), stating that the total energy consumption of ASM should not exceed the energy budget ϵ_s after changing execution cycles o_i to $o_i + \triangle o_i$. The T_i here is chosen as the upper-temperature limit in order to avoid thermal violations.
- **Timing Constraint** (12) states that the timing budget τ_s limits executing $o_i + \triangle o_i$ for each i.
- **Thermal Constraint** (13) is obtained from (8). It limits the temperature condition on the linear section of the piecewise leakage power model.

In this formulation, the variables of interest are the set of Δo_i and v_i for all ASM tasks i. The frequency f_i can only be determined after v_i is known, and is temporarily set as the lowest value for the time being. Note that it is not yet a QP formulation, due to the non-linear constraints (11) and (13). In order to linearize the constraints, we make two transformations: (1) Decouple the product form of (11) by substituting v_i to a prescribed value v_i^*; (2) Substitute $\Psi_1^f \vec{\mathrm{v}}^2 + \Psi_2^f \vec{\mathrm{v}}$ as an ensemble variable Θ. The above formulation is then transformed into the following QP formulation with variables of interests being Δo_i and Θ:
Minimize

$$\sum_i a_i((\frac{1}{2b_i^2}e^{-\frac{o_i}{b_i}})\triangle o_i^2 + (-\frac{1}{b_i}e^{-\frac{o_i}{b_i}})\triangle o_i) \tag{14}$$

Subject to

$$\sum_i (o_i+\triangle o_i)(C^{f_i}v_i^{*2}+K_\beta v_i^*+P_{i0,lkg}+K_\alpha T_i)f_i^{-1} \leq \epsilon_s \tag{15}$$

$$\frac{(o_i+\triangle o_i)}{f_i} \leq \tau_s \tag{16}$$

$$\vec{\mathrm{T}}^l \leq \Theta + \Phi^{f,T} \leq \vec{\mathrm{T}}^u \tag{17}$$

The problem (14)—(17) is a QP formulation and can be efficiently solved by the interior point method in polynomial time.

3.2 v_i^* Searching for Voltage Scaling

It would be ideal if the calculated v_i coincides v_i^* in (15). Otherwise, an iterative v_i^* adjustment process is needed to reduce the gap between v_i and v_i^* until zero. We consider iteratively solving the QP problem, during which process, v_i^* is reduced to the QP-solved v_i value if $v_i < v_i^*$. We show that the corresponding cycle Δo_i keeps increasing in this process.

To prove this concept, we assume that there are two consecutive rounds for QP optimization, namely r_1 and r_2. For task i, we set an initial voltage v_i^* of round r_1, then QP optimally produces voltage $v_i^{r_1}$ and $\triangle o_i^{r_1}$. Assume that $v_i^{r_1} < v_i^*$, we set $v_i^* = v_i^{r_1}$ as the initial voltage of round r_2, which results in new voltage $v_i^{r_2}$ and execution cycles $\triangle o_i^{r_2}$.

Theorem 1: For a given task i, if its voltages obtained during the iterative QP process has $v_i^{r_2} \leq v_i^{r_1} \leq v_i^*$, then $\triangle o_i^{r_2} \geq \triangle o_i^{r_1}$. In other words, when the voltage decreases between round r_1 and round r_2, $\triangle o_i$ is increasing.

Proof: We prove the claim by contradiction. Given that $<v_i^{r_1}, \triangle o_i^{r_1}>$ and $<v_i^{r_2}, \triangle o_i^{r_2}>$ are optimal solutions of round r_1 and round r_2, respectively, they should satisfy the energy and timing constraints. Assume $\triangle o_i^{r_2} < \triangle o_i^{r_1}$, and we have the following formulas for positive C^{f_i} and K_β:

$$\begin{aligned}(o_i+\triangle o_i^{r_2})(C^{f_i}{v_i^{r_1}}^2+K_\beta v_i^{r_1}) &< (o_i+\triangle o_i^{r_1})(C^{f_i}{v_i^{r_1}}^2+K_\beta v_i^{r_1}) \\ &< (o_i+\triangle o_i^{r_1})(C^{f_i}{v_i^*}^2+K_\beta v_i^*) \\ &\leq \epsilon_s^{'}\end{aligned}$$

and

$$o_i + \triangle o_i^{r_2} < o_i + \triangle o_i^{r_1} \leq \tau_s^{'}$$

where $\epsilon_s^{'}$ and $\tau_s^{'}$ are converted from (15) and (16). The above derivation shows that $\Delta o_i^{r_2}$ still has space to increase the value. Therefore, we can increase $\triangle o_i^{r_2}$ to a certain value, such as $\triangle o_i^{r_1}$, which still satisfies energy and timing constraints. The fact that $\triangle o_i^{r_2}$ can still be improved contradicts the assumption that it is the optimal solution in round r_2 in the QP formulation. Hence the proof. ■

3.3 The Overall DVFS Algorithm

Algorithm 1 depicts the overall voltage/frequency scaling process. Initially, all the v_i^* are set as the maximal value for further scaling down. After each round of *QP_Optimization* as described in Section 3.1, the task i with maximal v_i or maximal Δo_i improving potential is selected as the voltage scaling down candidate, and a new QP round is executed after setting $v_i^* = v_i$. This process continues until all $v_i \geq v_i^*$, and then we set all $v_i = v_i^*$ as the final voltage scaling results. The calculated Δo_i is treated as the optimal cycles obtained since they keep increasing, according to Theorem 1.

The frequency f_i can be determined after v_i values are known since v_i sets the upper limit for frequency scaling [20]. With frequency upscaling, there are still opportunities to improve Δo_i further, as shown in (16). However, energy and thermal constraints, namely (15) and (17), might be jeopardized. We adopt a trial-and-error approach to heuristically upscale the frequency to further maximize Δo_i: A random f_i is selected to upscale to the next discrete value $f_i^* \in F_{v_i}$ if (15) and (17) are not violated, and then upscaling is successful. Otherwise, choose the next task to repeat the trial process. The process ends until all f_i have been tried.

4 PARAMETER ESTIMATION FOR QUALITY FUNCTION

The key to DVFS optimization lies in the accurate parameter estimation of the quality function. Traditionally, the quality function parameters, such as a, b in (2), are obtained through extensive

Algorithm 1: VOLT/FREQ SCALING($\epsilon_s, \tau_s, \{i\}, \{F_{e,i}\}, T$)

```
for all i, v_i^* = v_max do
    {v_i, △o_i|∀i} = QP_Optimization();
    while ∃v_i < v_i^* do
        find i with MAX(v_i) or MAX(△o_i potential);
        v_i^* = v_i;
        {v_i, △o_i|∀i} ⇐ QP_Optimization();
    end
    while ∃v_i > v_i^* do
        v_i = v_i^*;
    end
    while {∃f_i ∈ F_{v_i}, ∀v_i} Not Tried do
        if Not Violate (15)(17) then
            f_i → f_i^*; Δo_i ⇐ τ_s * f_i^* − o_i;
            Update (15)(17);
        end
    end
end
return {Δo_i}
```

profiling and curve fitting [41]. However, during runtime, this method is impractical since the input patterns of the on-vehicle cameras are highly dynamic. In this work, our basic assumption is that vehicles could follow certain probability distribution to encounter a street, with someone traveled more frequently. For example, a household car may confine itself in the limited number of routes in daily life scenarios.[2] The input images taken into the system are thus semantically confined. Based on this simplification, we propose two estimation methods for parameters of the quality model:

- **Location similarity-triggered Feature Points Thresholding (L-FPT)**, which estimates the quality function based on historical estimation at the geo-location that exhibits high input similarity.
- **Street scenario-confined CNN (S-CNN)**, which is a neural network that trains and inferences the quality function that is used only for a specific street.

4.1 The L-FPT approach

The vehicle may repetitively come across the same geo-location, at which instance the input images, I_s, could exhibit high similarity to historical images, $I_{\hat{s}}$, taken at the same or nearby locations. In such cases, the characteristics of the processing loads are likely similar, thus could be attributed to similar quality functions for spatially recurred images.[3]

To evaluate the similarity of two images[4], namely the reference image and compared image, we resort to evaluating the *feature points matching rate*, R_{FP}, of the compared images. $R_{FP} \triangleq$

[2]There do exist counter-examples that fail this assumption (e.g., taxi cars). For such cases, we rely on user-level algorithmic solutions, such as transfer learning. Nonetheless, learning individual street's characteristics is still essential when applying to broader scenarios. In this paper, we focus on system-level computational optimization for the vehicular system.
[3]By "similar" we mean the exponential function parameters, $\langle a_i, b_i \rangle$, exhibit small variance in value.
[4]Without loss of generality, we compare the left images of the two evaluated stereo pairs.

$\frac{N_{comp}}{N_{ref}} \times 100\%$, where N_{ref} is the number of ORB feature points of the reference image, and N_{comp} is the number of feature points that can be found in both of the compared and reference images. The motivation behind this is that obtaining R_{FP} is an essential preprocessing step for state-of-the-art stereo matching algorithms, notably ELAS, where Sobel filtering is applied to generate the support points that are used for subsequent operations [10].

To obtain the feature points, off-the-shelf techniques such as Oriented FAST and Rotated BRIEF (ORB) [32], can be readily used without incurring significant overhead. An empirical threshold value for feature point matching rate, R_{FP}^{thresh}, is introduced to identify and admit spatially recurred images. Note that two images that have $R_{FP} \geq R_{FP}^{thresh}$ imply that they may slightly differ in horizontal, vertical, front, or back means. E.g., I_s could be taken at a slightly shifted vehicle position compared to $I_{\hat{s}}$, or at a location in front of or behind the $I_{\hat{s}}$ was taken. This considerably improves the practicality of L-FPT.

4.2 The CNN-based approach

If the I_s considered has $R_{FP} < R_{FP}^{thresh}$, the results of L-FPT could be inaccurate. In this case, we propose a CNN-based solution to estimate the parameters of the quality function.

4.2.1 S-CNN v.s. universal CNN. A straightforward design option is to use a universal CNN, which is a one-for-all network used to predict any single quality function given the stereo image input taken at any street/road section. However, this may cause issues such as slow weight update, which lags behind dynamically altering traffic context, thus leading to prediction inaccuracy as well as training inefficiency.

We propose a street scenario-confined CNN (S-CNN), which is trained by per-street data and used in that specific street as located by the vehicle. It improves efficiency and availability over the one-for-all CNN in the following aspects:

- Training speed and network size. Compared to the universal CNN, S-CNN requires limited training data, and the training can adapt to changes faster since the ground knowledge, namely the passive objects on the street, remains unchanged with high possibility. Although it should be avoided in general, the overfitting issue could potentially contribute to the training speed since under confined input training data, slightly enforcing overfitting helps "memorizing" rather than identifying the underlying distribution each time. The S-CNN can lead to a more concise network by removing superfluous variables and asking only relevant questions, which favors the embedded processing.
- Re-usability and self-containment. Rather than training a universal CNN for each vehicle, S-CNN is street-oriented such that different vehicles can share the same S-CNN whenever located on the street. Assuming trained in the cloud, the various vehicle could collaboratively contribute the training input images. Moreover, given the independence of the streets, a significant change in one street does not affect the S-CNNs of the other, thus ensuring the overall estimation performance of quality function during the vehicle's cruise activities.

4.2.2 ***Network architecture.*** We transform the problem of predicting quality function parameters into a classification problem, where the value range of the parameters is evenly and fine-grainedly divided. The output of the network is thus a prediction of the 'slot' that the parameters most likely fall in. We employ the LeNet network [19] for this purpose. The network has two layers of convolution and pooling operations, in addition to two fully connected layers. The output layer implements cross-entropy loss with softmax function to evaluate the output inconsistency [17]. Fig. 8 illustrates the architecture of the network proposed.

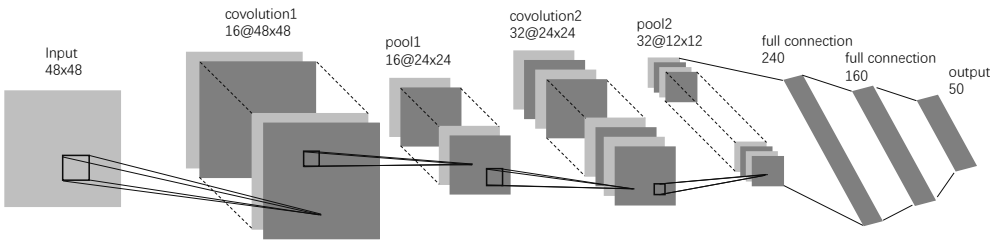

Fig. 8. LeNet network for S-CNN-based parameter prediction.

5 EVALUATION AND ANALYSIS

5.1 Experimental setup

We develop a simulated platform that consists of three computationally intensive modules, namely the DVFS module, the virtual processors, and the S-CNN inference module.

- **DVFS module** implements Algorithm 1, as well as algorithms for comparison. It intakes the quality functions and all energy, timing, and thermal constraints, and outputs the ASM execution cycles and voltage/frequency configurations. In our experiment, the maximum temperature is set at 70^oC, the initial frequencies of all the processors are set at 300MHz, and the initial energy budget is $0.04J$. Each ASM task has an exponential quality-cycle function, whose parameters are fed in from the S-CNN inference network or the L-FPT estimator. Empirically, typical values for the parameters are $a_i \in [0.2, 0.5]$ and $b_i \in [0.5, 1.2]$. The Matlab-based QP solver is employed to obtain the optimizing solution.
- **Virtual processors** are abstract mathematical modules that simulate running the ASM tasks. The processors are assumed to be 2x2 tiled with voltage/frequency scaling capability. Each core runs a simulated ASM task independently without preemption. The power and thermal characteristics of the processors are obtained from PTScalar [20]. The voltage of the processors ranges from 0.5V to 0.8V.
- **S-CNN inference module** is used to infer the parameters of the exponential quality functions. In the training phase, we train the S-CNN inference module by large datasets captured by Carla simulator [7]. Here we resize to 48*48 images due to vast datasets and set the output layer as 50 labels to divide the coefficient interval in detail. To obtain optimal accuracy, we set the training iteration number as 4000. The S-CNN is implemented using Tensorflow [1].

In our work, we test our algorithm with three datasets, namely Kitti [9], New Tsukuba [24], and Carla [7]. Kitti and New Tsukuba are two widely used datasets to evaluate the stereo matching algorithm. Carla dataset is obtained from the Carla simulator to support the development, training, and validation of smart vehicular systems [7]. To verify the performance of our algorithm, we have compared it with three state-of-the-art algorithms: (1) **TS** — a scheduling heuristic that maximizes the ASM execution cycles by selecting the task with the highest energy metrics [45]. (2) **DFS** — a scheduling method to optimize the quality with the exponential model through dynamic frequency scaling [41]. (3) **ED** — a scheduling method is identical to our proposed approach, except that it evenly distributes the energy budget to all the processors.

5.2 Results and Analysis

5.2.1 Accurate quality function modeling. In this part, we evaluate the accuracy of using exponential functions to model quality-cycle relationships. We compare exponential models with linear and power models. Sample images are randomly selected from the three datasets [7, 9, 24], and the

curve fitting technique is used to examine the modeling errors. The modeling error is evaluated using the Mean Square Error (MSE) between the original quality values and the fitted values. Fig. 9 illustrates the modeling results of randomly selecting 20 sample images from each of the three datasets. The MSEs are in the magnitude of 10^{-1}, 10^{-2}, and 10^{-3} for linear, power, and exponential modeling, respectively. This indicates a generally smaller modeling error using the standard exponential models, as defined in (2).

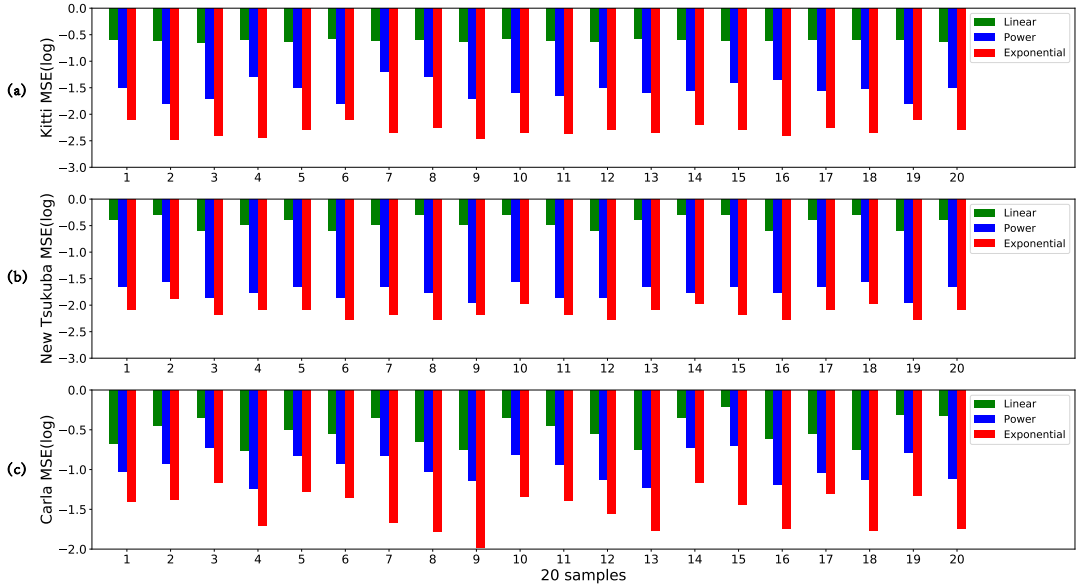

Fig. 9. Comparing MSE of three different models (exponential, power, and linear) using 20 random samples from three datasets (top to bottom - Kitti, New Tsukuba, Carla). MSE values are taken logarithms for better illustration.

5.2.2 Performance of DVFS. We evaluate the efficacy of our proposed DVFS algorithm on the 60 randomly selected images in the previous experiment. Table 2 lists the averaged results of applying our approach, including the improvement of disparity/quality and execution cycles, as well as optimized system voltage and energy consumption. Execution results are compared with the initial disparity (d_{max}) configurations, given as 62, 72, and 38, respectively. Initial voltages are all set as $0.8v$. The third row of each group of tests shows the optimal results after applying our approach, where the disparity ranges increase to 197, 203, and 59, respectively, for images form each dataset. The last column of Table 2 shows that our approach achieves 2.21, 1.93, and 1.60 times overall quality improvement on the three datasets, respectively.

We also pick sample tests where the execution cycles are roughly at the middle between the initial and optimized disparity settings (50% in the range), as shown in the second row of each group of tests. It is interesting to observe that: (1) The disparity value is also in the middle of the range, obeying the linear relationship between the disparity and execution cycles, as shown in Fig. 6. (2) The output quality may not linearly increase with execution cycles. Rather, it improves less quickly as the cycle increases. Observed from Table 2, it shows that the quality improvement ratios before and after the middle cycle increase point (namely second row) are <82.1%, 17.9%>,

<76.9%, 23.1%>, and <74.3%, 25.7%>, respectively for the three datasets. This validates our strategy of directly optimizing the ASM quality rather than the ASM execution cycles since quality does not increase uniformly with execution cycles for ASM tasks.

Table 2. ASM output quality under disparity ranges, execution cycles, and voltages calculated from our proposed DVFS algorithm on three datasets. Please refer back to Fig. 5 to visualize the matching quality improvement in the three datasets.

Datasets	Disparity(pixels)	Cycles(billion)	Voltage(V)	Energy(J)	Quality
Kitti	62	0.0914	0.8	0.0109	3.33
	128	0.2114 (mid)	0.5934	0.0124	7.11
	197	0.3387	0.5856	0.0249	**7.36**
Newtsukuba	72	0.1315	0.8	0.0127	3.57
	140	0.2552 (mid)	0.5933	0.0136	6.46
	203	0.3714	0.5786	0.025	**6.89**
Carla	38	0.0694	0.8	0.0052	2.28
	49	0.0906 (mid)	0.5945	0.0067	3.29
	59	0.1083	0.5844	0.0073	**3.64**

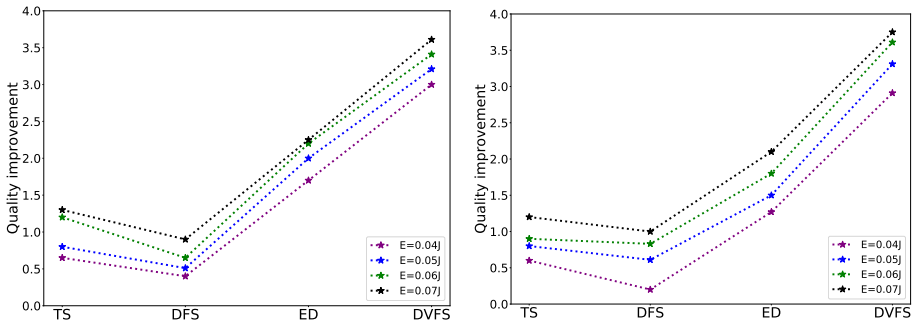

Fig. 10. Comparison of quality improvement using **TS**, **DFS**, **ED**, and **DVFS**, under different energy budgets. Left: the results averaged from evaluating the straight road section images. Right: the results obtained from evaluating the road junction images, all obtained from the Carla simulator.

Fig. 10 shows the results of comparing our **DVFS** approach with the **TS**, **DFS**, and **ED** approaches. The results are the average values of executing the algorithms using 1000 straight road section images and 1000 road junction images captured from Carla, under various energy constraints. Results show that on the images of straight road sections, our approach achieves on average 1.61× more quality over **ED**, 2.78× over **TS**, and 4× over **DFS**. On the images of road junctions, our approach achieves on average 1.79× more quality over **ED**, 3.13× over **TS**, and 3.75× over **DFS**. In both cases, the advantage over **ED** is due to optimally distributing the energy resource among the processors through (15), rather than rigidly distributing the energy budget. **TS** distributes the execution cycles prioritized by the energy coefficient, rather than directly optimizing the quality. **DFS** only considers frequency scaling under the voltage of 0.8V. Although this voltage gives maximal frequency scaling range, only considering frequency scaling is less effective for quality

improvements. This is due to ignoring the significant role that voltage plays in dynamic+leakage energy consumption.

Fig. 11 numerically shows the inaccuracy of the quality metric introduced by using Taylor's expansion. The value of each column indicates the differences of the quality obtained from the optimization computation, which adopts Taylor's expansion, and the quality calculated from the cycle input according to the exponential model. Over the 20 samples examined, the maximum quality error due to Taylor's expansion is at a scale of 10^{-3}.

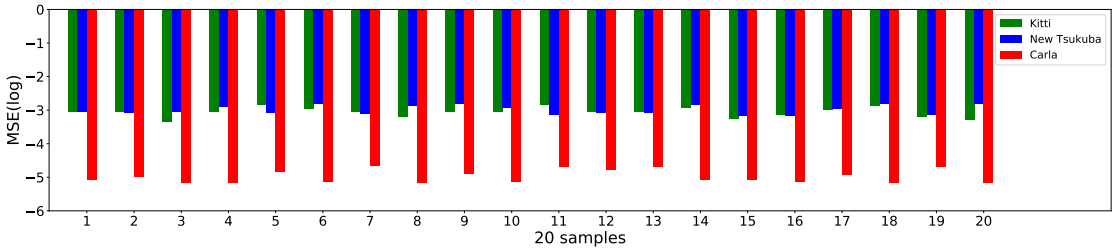

Fig. 11. Comparing MSE of actual exponential cost with Taylor expanded approximation using 20 random samples from three datasets. MSE values are taken logarithms for better illustration.

5.2.3 Effectiveness of L-FPT. Recall that L-FPT assumes that at a certain location (could be with slight position shift), the processor executing ASM tasks with such inputs exhibits similar quality-cycle relationships; thus the parameters can be guessed from historical records at the same location. To validate this assumption, we randomly select a fixed location in the Carla simulator, took a reference image as shown in Fig. 12(a), and manually creates 300 images with increased density of vehicles and pedestrians (see samples in Fig. 12(b)-(d)). The purpose is to alter the feature points matching rate R_{FP} such that within a certain threshold R_{FP}^{thresh}, the function parameters have small differences in value. Fig. 13(a) shows that with $R_{FP} \geq 85\%$, which is empirically obtained for the considered location, the differences of parameters $|a_i - a_j|$ and $|b_i - b_j|$ are within 3.38% and 3.19%, respectively, compared with the worst-case differences in Fig. 13(a). Correspondingly, Fig. 13(b) shows that with $R_{FP} \geq 85\%$, the ASM quality difference is confined within 7.15%.

(a) (b) (c) (d)

Fig. 12. Illustration of images taken at the selected location, with increasing density of pedestrians and vehicles. The feature point matching rate increases accordingly. (a) reference image; (b) $R_{FP} = 93\%$; (c) $R_{FP} = 88\%$; (d) $R_{FP} = 81\%$.

Fig. 14 reflects a scenario that the vehicle is turning around a road junction, and illustrates the changes in the function parameters (a_i, b_i) and the calculated disparity range during this process.

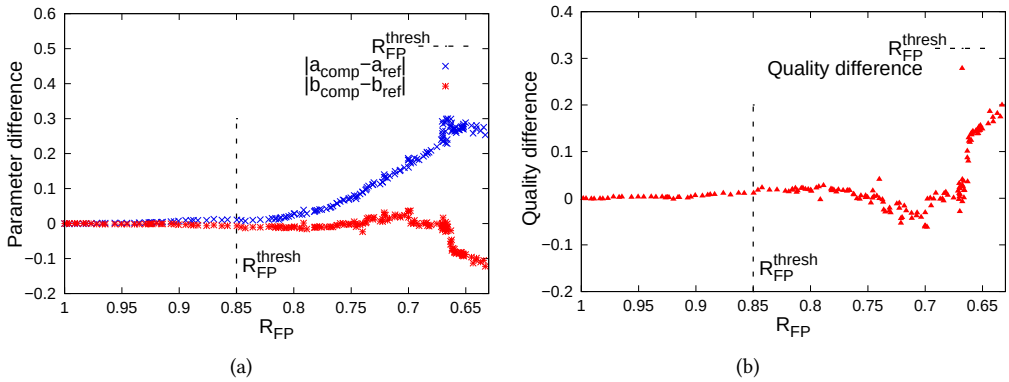

Fig. 13. Feature point analysis at one location under the different density of vehicles and pedestrians. Left: Low parameter differences when the R_{FP} is below a threshold. Right: Low output quality difference when the R_{FP} is below the threshold.

This process is captured by a continuous video clip of 28 seconds with framerate 30fps. The parameters (a_i, b_i) of each frame is profiled as in Fig. 7. The corresponding disparity range is calculated using our proposed DVFS algorithm, where the settings follow Section 5.2.2. Two scenarios are studied for the applicability of L-FPT. As can be observed from points (a)(b)(c) in Fig. 14, which correspond to frames 160, 180, and 200, and reflect a certain location with a slight front/back shift. At this location, values of a_i and b_i fluctuate in a small range, while the calculated disparity remains constant in the meantime. This implies that, to obtain the same calculated disparity range (hence the output quality), a_i and b_i values can tolerate certain fluctuations and inaccuracy. Thus, it is possible to avoid calculating a_i and b_i for each frame during this process but using the predefined historical values that satisfy the L-FPT requirement. This could help avoid a considerable computation workload. On the other hand, points at (d)(e)(f) in Fig. 14 correspond to frames 580, 610, and 650, and reflect the process of turning at the road junction. The L-FPT approach would fail in this process since the images taken before and after the junction turning could be quite different. Fig. 15 shows the corresponding images, as indicated in Fig. 14.

5.2.4 Effectiveness of S-CNN. In order to verify the proposed S-CNN approach, in the Carla simulator, we collect images captured during traversing straight sections of ten different streets, as well as ten street junctions, respectively, illustrated in Fig. 16. For each street, we randomly collect 14.4K images with different densities of pedestrians or vehicles. We divide the 14.4K images into the training set of 12K images and the test set of 2.4K images. The output layer used to predict the model coefficient is divided into 50 intervals, and the training iteration is 4000. To train and test the universal CNN, we capture 500K images with different densities of pedestrians or vehicles over the whole simulated town. In this dataset, the portions of the straight sections and junctions are even. We use 492.5K images to train and 7.5K images to test the performance of the universal CNN approach.

Table 3 and 4 show the MSE and accuracy results of employing the S-CNN and universal CNN methods for estimating the parameters of the exponential quality models. Several findings are detailed here: (1) The accuracy of predicting the junction images (T_x) using S-CNN is lower compared to predicting the straight section images (S_x). Specifically, for the average-10 (T-avg10 and S-avg10) results, the prediction accuracy of T-avg10 is 17.2% and 14.1% less than that of S-avg10 for a_i and b_i, respectively. The MSE data agrees with this conclusion, indicating that S-CNN works

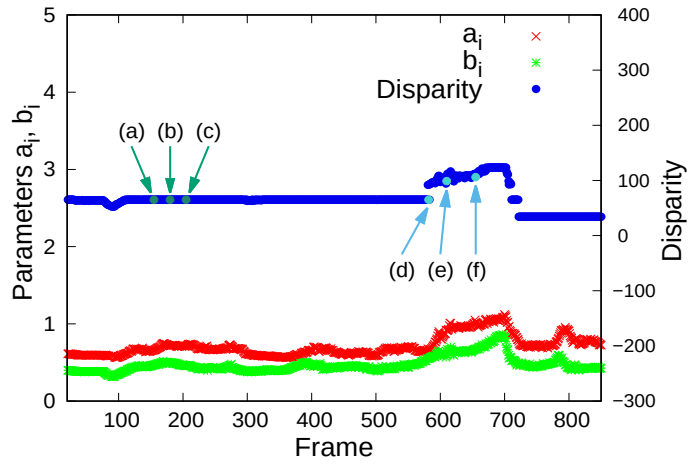

Fig. 14. Parameters and optimal disparity obtained in the process of junction turning. Instances (a)-(f) corresponds to the images in Fig. 15.

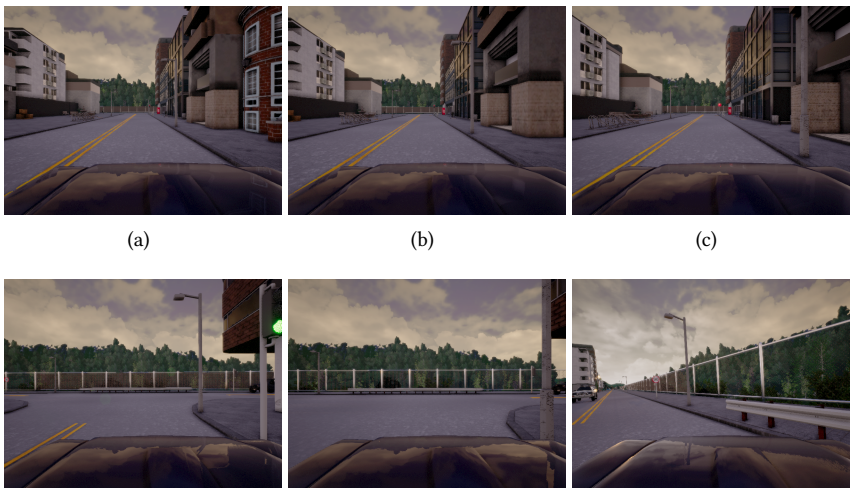

Fig. 15. Instances of the vehicle turning a road junction. (a)-(c) represent continuous scenes with high location similarity. (d)-(f) represent continuous scenes scenarios with low location similarity, before and after turning the junction.

well on straight sections, but less effective when the vehicle approaches the junctions. This is expected since the junctional image sets contain nearly half of the images taken from the street after turning around the junction. That street is not the target of the S-CNN trained specifically for the street before the junction. (2) After adopting the hybrid dataset that consists of straight sections and junction images, the average accuracy of the universal CNN is 11.4% less than the S-CNN on straight section-only scenarios, but 4.26% higher than the S-CNN on junction-only scenarios.

Regression-based methods have also been well recognized for parameter prediction. Thus, we design experiments to evaluate the prediction accuracy of the regression-based methods as compared to S-CNN. We define two types of predictors related to the stereo matching workload: (1) The number of matched feature points; and (2) the *Bhattacharyya Coefficient* (B. C.) [29] that is used to measure the similarity of two images, which are represented by histograms on the number of pixels that fall in the numerical range of the greyscale. Four algorithms are employed to obtain the number of matched feature points, namely SURF, SIFT, ORB, and AKAZE, available from the OpenCV library [3]. We use 10K images for regression model training. As shown in Table 5, the Fourier regression with the B. C. predictor exhibits the best MSE. We supply the Fourier regression results into Table 3 and 4 to make comparison with the CNN-based approaches. Results show that the accuracy of the regression-based method is on average 69.6% lower than the S-CNN based one.

Table 3. Comparison of MSE using S-CNN, Regression, Universal CNN approaches.

	S-CNN												
Param	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T-avg10		
a_i	0.0823	0.0833	0.0801	0.0805	0.081	0.0806	0.0826	0.0802	0.0796	0.0808	0.0811		
b_i	0.0512	0.0533	0.0503	0.0487	0.0495	0.0515	0.0492	0.0534	0.0536	0.0533	0.0514		
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S-avg10	Regr.	U-CNN
a_i	0.0675	0.0664	0.0643	0.065	0.0641	0.0678	0.0662	0.0647	0.0667	0.0663	0.0659	0.1027	0.0757
b_i	0.0381	0.0375	0.0368	0.0372	0.0372	0.0379	0.0371	0.0377	0.0385	0.038	0.0376	0.0816	0.0499

Table 4. Comparison of accuracy(%) using S-CNN, Regression, Universal CNN approaches.

	S-CNN												
Param	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T-avg10		
a_i	77.3	84	79.2	78.2	77.1	82.07	78.4	80.2	77.13	77.3	79.09		
b_i	80.6	83.2	83.9	83.1	80.1	81.4	81.8	82	83.04	83.86	82.3		
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S-avg10	Regr.	U-CNN
a_i	94.42	96.05	96.72	96.9	96.75	96.2	96.02	96.3	96.45	97.09	96.29	24.5	85.7
b_i	95.6	96.44	96.43	96.2	96.12	96.41	96.22	96.4	97.33	96.95	96.41	29	84.2

(a) (b) (c) (d)

Fig. 16. Illustration of images taken at Str1, Str2, Turn1, and Turn2, respectively.

Table 5. The results of five different regression methods with predictors chosen as Bhattacharyya Coefficient, as well as the number of matched feature points employing SURF, SIFT, ORB, and AKAZE.

Models	Param	MSE				
		Feature Points				B. C. (Histogram)
		SUFR	SIFT	ORB	AKAZE	
Linear	a_i	0.1221	0.1225	0.1222	0.1213	0.1085
	b_i	0.0948	0.0952	0.0949	0.0939	0.0839
Fourier	a_i	0.1171	0.1096	0.1187	0.1170	**0.1027**
	b_i	0.0899	0.0871	0.0913	0.0902	**0.0816**
Exponential	a_i	0.1194	0.1162	0.1203	0.1189	0.1185
	b_i	0.0920	0.0903	0.0931	0.0916	0.0938
Polynomial	a_i	0.1199	0.1193	0.1206	0.1192	0.1183
	b_i	0.0928	0.0928	0.0935	0.0920	0.0937
Power	a_i	0.1164	0.1066	0.1144	0.1133	0.1195
	b_i	0.0898	0.0863	0.0896	0.0884	0.0943

Table 6. Execution time of S-CNN, ASM, ED, and DVFS tasks on FPGA implementation. Corresponding quality improvement under the ED and our DVFS is also shown. E.g., Quality× = 1.68 indicates 1.68 times of quality improvement.

		Str1	Str2	Str3	Str4	Str5	Turn1	Turn2	Turn3	Turn4	Turn5	Average
S-CNN	Time(ms)	0.5151	0.5221	0.5187	0.5247	0.5431	0.5581	0.5484	0.5231	0.5216	0.5444	0.5319
ASM	Time(ms)	26.2179	27.4511	28.0501	25.6419	24.4362	27.0473	27.5902	26.4196	26.4146	25.3292	26.4598
ED	Time(ms)	0.122	0.128	0.098	0.119	0.116	0.138	0.131	0.118	0.132	0.134	0.1236
	Quality×	1.68	1.71	1.70	1.68	1.72	1.23	1.19	1.21	1.22	1.20	1.454
DVFS	Time(ms)	0.124	0.132	0.103	0.122	0.119	0.143	0.135	0.123	0.135	0.137	0.1273
	Quality×	3.02	3.15	3.10	3.01	3.23	2.90	2.70	2.81	2.86	2.77	2.955

5.2.5 Timing analysis. The proposed framework consists of several important components, including the ASM (ELAS) algorithm, CNN for parameter inference, and the DVFS algorithm. To reflect its feasibility on realistic on-vehicle systems, we deploy the three components on FPGA for fast validation, using Xilinx UltraScale+ ZCU102 with the SDSoC framework. We randomly select 10 sample images of the size 1275×375 captured from Carla. In addition, we use the same window sizes for ELAS algorithms, i.e., 9×9 and 5×5, respectively, and set the number of support points as 32 [31]. To facilitate comparison, the clock frequency of all the accelerated parts is set to 300MHz. The QP core is the central functionality of the DVFS algorithm. We adopt the reference implementation of the QP solver from [16], while the external control logics are implemented using SDSoC. The reference FPGA design of the S-CNN is adopted from [46]. In Table 6, the first row shows the execution time of S-CNN inference for corresponding parameters. The second row shows the execution time of randomly selected 10 ASM tasks. The rest of the rows show the DVFS and ED approaches' execution time to decide the optimal cycles, as well as their corresponding quality improvement. The time of DVFS and S-CNN inference is negligible compared to the ELAS algorithm. Compared to the ED approach where the energy constraint is simplified, the average

quality improvement of our DVFS method is about 2.03× over the ED method, incurring only 2.9% more timing consumption compared to ED.

6 CONCLUSION

In this paper, we convert a stereo matching algorithm into an ASM that can be used for smart vehicle platforms, derive an efficient DVFS algorithm to maximize the ASM's output quality under timing, energy, and thermal constraints, and devise L-FPT and S-CNN methods to estimate the parameters of the ASM quality function accurately. The work is the first kind that studies the application-level adaptability for smart vehicular systems. Results show that our approach achieves at least 1.61 times quality improvement compared to contemporary techniques, and the average parameter estimation method achieves 96.35% accuracy on the straight road. In the future, we will integrate the FPGA-based system onto our driverless car prototype, and evaluate the system under realistic conditions.

ACKNOWLEDGEMENT

The research is supported by The Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning.

REFERENCES

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. 265–283. https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[2] H. Aydin, R. Melhem, D. Mosse, and P. Mejia-Alvarez. 2001. Optimal reward-based scheduling for periodic real-time tasks. *IEEE Trans. Comput.* 50, 2 (Feb 2001), 111–130. https://doi.org/10.1109/12.908988

[3] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[4] Vinay K. Chippa, Kaushik Roy, Srimat T. Chakradhar, and Anand Raghunathan. 2013. Managing the Quality vs. Efficiency Trade-off Using Dynamic Effort Scaling. *ACM Trans. Embed. Comput. Syst.* 12, 2s, Article 90 (May 2013), 23 pages. https://doi.org/10.1145/2465787.2465792

[5] J. . Chung, J. W. S. Liu, and K. . Lin. 1990. Scheduling periodic jobs that allow imprecise results. *IEEE Trans. Comput.* 39, 9 (Sep. 1990), 1156–1174. https://doi.org/10.1109/12.57057

[6] R. Danescu, F. Oniga, and S. Nedevschi. 2011. Modeling and Tracking the Driving Environment With a Particle-Based Occupancy Grid. *IEEE Transactions on Intelligent Transportation Systems* 12, 4 (Dec 2011), 1331–1342. https://doi.org/10.1109/TITS.2011.2158097

[7] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. 1–16.

[8] A. Ess, B. Leibe, K. Schindler, and L. van Gool. 2009. Robust Multiperson Tracking from a Mobile Platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 10 (Oct 2009), 1831–1846. https://doi.org/10.1109/TPAMI.2009.109

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074

[10] Andreas Geiger, Martin Roser, and Raquel Urtasun. 2011. Efficient Large-Scale Stereo Matching. In *Computer Vision – ACCV 2010*, Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 25–38.

[11] V. Hanumaiah and S. Vrudhula. 2012. Temperature-Aware DVFS for Hard Real-Time Applications on Multicore Processors. *IEEE Trans. Comput.* 61, 10 (Oct 2012), 1484–1494. https://doi.org/10.1109/TC.2011.156

[12] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. 2013. Fast Cost-Volume Filtering for Visual Correspondence and Beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 2 (Feb 2013), 504–511. https://doi.org/10.1109/TPAMI.2012.156

[13] Albert S. Huang, Abraham Bachrach, Peter Henry, Michael Krainin, Daniel Maturana, Dieter Fox, and Nicholas Roy. 2011. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *ISRR*. 466–474. https://doi.org/10.1109/CVPR.2015.7298644

[14] H. Huang, V. Chaturvedi, G. Quan, J. Fan, , and M. Qiu. 2014. Throughput Maximization for Periodic Real-time Systems Under the Maximal Temperature Constraint. *ACM Trans. Embed. Comput. Syst.* 13, 2s, Article 70 (Jan. 2014), 22 pages. https://doi.org/10.1145/2544375.2544390

[15] Wei Huang, Shougata Ghosh, Sivakumar Velusamy, Karthik Sankaranarayanan, Kevin Skadron, and Mircea Stan. 2006. HotSpot: A compact thermal modeling methodology for early-stage VLSI design. *IEEE Transactions on Very Large Scale Integration Systems - VLSI* 14 (05 2006), 501–513. https://doi.org/10.1109/TVLSI.2006.876103

[16] Juan Luis Jerez, George A. Constantinides, and Eric C. Kerrigan. 2011. An FPGA implementation of a sparse quadratic programming solver for constrained predictive control. In *FPGA*.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*. 1097–1105.

[18] Kuk-Jin Yoon and In So Kweon. 2006. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 4 (April 2006), 650–656. https://doi.org/10.1109/TPAMI.2006.70

[19] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (Nov 1998), 2278–2324. https://doi.org/10.1109/5.726791

[20] Weiping Liao, Lei He, and Kevin M. Lepak. 2005. Temperature and supply Voltage aware performance and power modeling at microarchitecture level. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 24 (08 2005), 1042 – 1053. https://doi.org/10.1109/TCAD.2005.850860

[21] Jiling Liu, Yong Zhang, and Xueguang Dong. 2015. Local stereo matching based on the improved matching cost function and the adaptive window. In *2015 8th International Congress on Image and Signal Processing (CISP)*. IEEE, 287–292.

[22] Tong Liu, Xiyuan Peng, and Qiao Li-yan. 2016. Window-Based Three-Dimensional Aggregation for Stereo Matching. *IEEE Signal Processing Letters* 23 (07 2016), 1–1. https://doi.org/10.1109/LSP.2016.2578944

[23] Frank D. Macías-Escrivá, Rodolfo Haber, Raul del Toro, and Vicente Hernandez. 2013. Self-adaptive systems: A survey of current approaches, research challenges and applications. *Expert Systems with Applications* 40, 18 (2013), 7267 – 7279. https://doi.org/10.1016/j.eswa.2013.07.033

[24] Sarah Martull, Martin Peris, and Kazuhiro Fukui. 2012. Realistic CG stereo image dataset with ground truth disparity maps. In *ICPR workshop TrakMark2012*, Vol. 111. 117–118.

[25] Lei Mo, Angeliki Kritikakou, and Olivier Sentieys. 2017. Decomposed Task Mapping to Maximize QoS in Energy-Constrained Real-Time Multicores. In *2017 IEEE International Conference on Computer Design (ICCD)*. 493–500. https://doi.org/10.1109/ICCD.2017.86

[26] Andrew Nelson, Benny Akesson, Anca Molnos, Sjoerd Te Pas, and Kees Goossens. 2012. Power versus quality trade-offs for adaptive real-time applications. In *2012 IEEE 10th Symposium on Embedded Systems for Real-time Multimedia*. IEEE, 75–84.

[27] O. Ozturk, M. Kandemir, and G. Chen. 2013. Compiler-Directed Energy Reduction Using Dynamic Voltage Scaling and Voltage Islands for Embedded Systems. *IEEE Trans. Comput.* 62, 2 (Feb 2013), 268–278. https://doi.org/10.1109/TC.2011.229

[28] Sylvain Paris, Pierre Kornprobst, Jack Tumblin, Frédo Durand, et al. 2009. Bilateral filtering: Theory and applications. *Foundations and Trends® in Computer Graphics and Vision* 4, 1 (2009), 1–73.

[29] Bidyut Kr. Patra, Raimo Launonen, Ville Ollikainen, and Sukumar Nandi. 2015. A New Similarity Measure Using Bhattacharyya Coefficient for Collaborative Filtering in Sparse Data. *Know.-Based Syst.* 82, C (July 2015), 163–177. https://doi.org/10.1016/j.knosys.2015.03.001

[30] M. Perrollaz, J. Yoder, A. Negre, A. Spalanzani, and C. Laugier. 2012. A Visibility-Based Approach for Occupancy Grid Computation in Disparity Space. *IEEE Transactions on Intelligent Transportation Systems* 13, 3 (Sep. 2012), 1383–1393. https://doi.org/10.1109/TITS.2012.2188393

[31] O. Rahnama, D. Frost, O. Miksik, and P. H. S. Torr. 2018. Real-Time Dense Stereo Matching With ELAS on FPGA-Accelerated Embedded Devices. *IEEE Robotics and Automation Letters* 3, 3 (July 2018), 2008–2015. https://doi.org/10.1109/LRA.2018.2800786

[32] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*. 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544

[33] Cosmin Rusu, Rami Melhem, and Daniel Mossé. 2003. Maximizing Rewards for Real-time Applications with Energy Constraints. *ACM Trans. Embed. Comput. Syst.* 2, 4 (Nov. 2003), 537–559. https://doi.org/10.1145/950162.950166

[34] Daniel Scharstein and Richard Szeliski. 2002. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision* 47, 1 (01 Apr 2002), 7–42. https://doi.org/10.1023/A:1014573219977

[35] Daniel Scharstein and Richard Szeliski. 2003. High-Accuracy Stereo Depth Maps Using Structured Light. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*. IEEE Computer Society, USA, 195–202.

[36] Stephan Schraml, Ahmed Nabil Belbachir, and Horst Bischof. 2015. Event-driven stereo matching for real-time 3D panoramic vision. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), 466–474.

[37] Wenjie Song, Yi Yang, Mengyin Fu, Yujun Li, and Meiling Wang. 2018. Lane Detection and Classification for Forward Collision Warning System Based on Stereo Vision. *IEEE Sensors Journal* 18, 12 (June 2018), 5151–5163. https://doi.org/10.1109/JSEN.2018.2832291

[38] R. Thakur. 2016. Scanning LIDAR in Advanced Driver Assistance Systems and Beyond: Building a road map for next-generation LIDAR technology. *IEEE Consumer Electronics Magazine* 5, 3 (July 2016), 48–54. https://doi.org/10.1109/MCE.2016.2556878

[39] Sihan Wen. 2017. Convolutional neural network and adaptive guided image filter based stereo matching. In *IEEE IST*. 1–6. https://doi.org/10.1109/IST.2017.8261530

[40] Heng Yu, Yajun Ha, and Bharadwaj Veeravalli. 2013. Quality-Driven Dynamic Scheduling for Real-Time Adaptive Applications on Multiprocessor Systems. *IEEE Trans. Comput.* 62, 10 (2013), 2026–2040. https://doi.org/10.1109/TC.2012.194

[41] Heng Yu, Yajun Ha, and Jing Wang. 2017. Quality Optimization of Resilient Applications Under Temperature Constraints. In *Proceedings of the Computing Frontiers Conference (CF'17)*. ACM, New York, NY, USA, 9–16. https://doi.org/10.1145/3075564.3075577

[42] Heng Yu, Bharadwaj Veeravalli, Yajun Ha, and Shaobo Luo. 2013. Dynamic Scheduling of Imprecise-Computation Tasks on Real-Time Embedded Multiprocessors. In *2013 IEEE 16th International Conference on Computational Science and Engineering*. 770–777.

[43] Q. Zhang, F. Yuan, R. Ye, and Q. Xu. 2014. ApproxIt: An Approximate Computing Framework for Iterative Methods. In *Proc. Design Automation Conf.* 1–6.

[44] Sushu Zhang and Karam S Chatha. 2010. Thermal aware task sequencing on embedded processors. In *Proceedings of the 47th Design Automation Conference*. ACM, 585–590.

[45] Junlong Zhou, Jianming Yan, Tongquan Wei, Mingsong Chen, and Xiaobo Sharon Hu. 2017. Energy-Adaptive Scheduling of Imprecise Computation Tasks for QoS Optimization in Real-Time MPSoC Systems. https://doi.org/10.23919/DATE.2017.7927212

[46] Yongmei Zhou and Jingfei Jiang. 2015. An FPGA-based accelerator implementation for deep convolutional neural networks. *2015 4th International Conference on Computer Science and Network Technology (ICCSNT)* 01 (2015), 829–832.