Towards Situation-Aware Meta-Optimization of Adaptation Planning Strategies

Veronika Lesch
University of Würzburg
Würzburg, Germany
veronika.lesch@uni-wuerzburg.de

Tanja Noack
University of Hohenheim
Stuttgart, Germany
tanja.noack@uni-hohenheim.de

Johannes Hefter
University of Würzburg
Würzburg, Germany
johannes.hefter@informatik.uni-wuerzburg.de

Samuel Kounev
University of Würzburg
Würzburg, Germany
samuel.kounev@uni-wuerzburg.de

Christian Krupitzer
University of Hohenheim
Stuttgart, Germany
christian.krupitzer@uni-hohenheim.de

***Abstract*—Platooning is a promising approach for optimizing the usage of the existing road infrastructure by driving in convoys with low inter-vehicle distances. Platooning coordination fosters advantages like an increased road throughput and reduced fuel consumption. Diverse so-called platooning coordination strategies exist in the literature and according to the no-free-lunch theorem, each has individual assets and drawbacks, making them best applicable for different traffic situations.**

This paper proposes a layered system model and a feedback loop for meta-optimization of self-adaptive systems. We apply our concept on platooning coordination as a case study and provide an experience report. The platooning coordination strategy is exchangeable and its input parameters are tuned to fit the current traffic situation. Our evaluation results show that the choice of the platooning coordination strategy is situation-dependent. Further, our meta-optimization of the input parameters of these strategies for the traffic situation is favorable compared to a static approach.

***Index Terms*—Situational Awareness, Meta-Optimization, Platooning Coordination**

I. Introduction

The amount of traffic on the roads increases every year; for example, in Germany, an increase in the total vehicle stock by approximately one million vehicles was observed from 2018 to 2019 [1]. Authorities try to tackle this issue by expanding the road network, which leads to increased costs [2]. Due to the advancement in autonomous driving, this infrastructure demand can be reduced through platooning, which is the ability of vehicles to drive at very low inter-vehicle distances enabled by communication [3]. Utilizing platooning, the road throughput [4] and safety [3] increases while driving.

While the feasibility of platooning is shown in diverse projects, the issue of platooning coordination still exists. Platooning coordination is the process of assigning vehicles to platoons and controlling the platooning activities. The platooning coordination problem is a multi-objective problem with multiple dimensions since objectives of the drivers, aspects of the platoon and global traffic need to be considered [5] as well as fairness between participants must be guaranteed as the leading vehicle benefits less from slipstream effects [6].

Following the observation from [7] that the choice of the algorithm for adaptation planning in self-adaptive systems [8], [9], is dependent on the situation of the system, we claim that the choice of the platooning coordination strategy also is situation-dependent. However, it will not be feasible to define the best-fitting strategy for each traffic situation determined by various parameters such as the number of vehicles, the ratio of trucks versus cars, or the number of lanes [10]. Accordingly, we propose to define a matching of strategies for everyday traffic situations and apply a meta-optimization of the strategy's parameters to fit them to the specific traffic situation. We analyze different strategies and optimization algorithms under varying traffic situations to show the usefulness of combining a situation-dependent choice of the adaptation planning strategy with a meta-optimization of the parameters. Following this adaptation approach, this work improves the platooning coordination process by adapting an existing system via selecting a platooning coordination strategy and optimizing the strategy's configuration based on the current traffic situation. Hence, our contribution is threefold:

- We define a 3-layered system model for meta-optimization in self-adaptive systems using the example domain of platooning coordination.
- By analyzing a set of platooning coordination strategies and optimizing them, we provide a multi-objective meta-optimized platooning coordination process.
- We propose a customizable and reusable testbed for evaluating meta-optimized adaptation planning algorithms.

The remainder of the paper is structured as follows. Section II discusses related works. Section III proposes our system model, our feedback loop adaptation and use case specific details. Section IV defines details on platooning coordination and Section V presents used optimization algorithms. Section VI presents our reusable evaluation testbed before Section VII summarizes our evaluation results and discusses threats to validity. Section VIII discusses upcoming challenges regarding our vision and Section IX concludes the paper.

II. Related Work

Neumüller et al. [11] present a parameter meta-optimization implementation for the heuristic optimization environment *HeuristicLab Hive*. Feurer et al. [12] improve the Sequential Model-based Bayesian Optimization (SMBO) used for tuning the parameters of machine learning algorithms integrating meta-learning. Chis et al. [13] analyze the design space exploration process using the framework for automatic design space exploration (FADSE) to compare the performance of various multi-objective meta-heuristics. Similarly, Vinctan et al. [14] address design space exploration by implementing a meta-optimization layer for the FADSE tool.

According to Lewis et al. [15], meta-self-awareness "*leads to the ability to model and reason about changing trade-offs during the system's lifetime*". Gerostathopoulos et al. [16] propose the concept of meta-adaption for cyber-physical systems, which improves the adaption of a cyber-physical system by generating new self-adaption strategies at runtime. Kinneer et al. [17] propose an approach based on a genetic algorithm for reusing historical knowledge of existing plans for adaptation planning.

A recent study from Calinescu et al. [18] has shown that situation-awareness is the key driver for developing self-adaptive systems and, hence, is still an important research topic with many open research challenges. Fredericks et al. [7] propose a concept that enables a system to identify the situation it is in and detect the optimal system configuration for this situation at runtime. Hardes et al. [19] address communication issues in urban platooning scenarios by utilizing the concept of situation-awareness. Porter et al. [20] propose a software framework that learns optimal system assemblies in emergent software systems. Kang et al. [21] analyze which history length and sensor range yields the best results on long-term situational awareness.

This work delineates from the presented related work as follows: The first presented meta-optimization approaches mainly focus on optimizing machine learning or design space exploration techniques. In contrast, we apply meta-optimization on adaptation planning strategies for adaptive systems. Further, the meta-self-awareness approaches focus on the generation of new adaptation strategies or the use of historical knowledge to plan adaptations at runtime while we depend on the existing strategies and aim at optimizing the input parameters of these strategies. The approaches on situational awareness inspired us to use an optimization algorithm for situation-dependent parameter optimization. Hence, we implement optimization approaches and compare their performance in our use case. Finally, to the best of our knowledge, no other work proposes meta-optimization of platooning coordination what makes our work a unique contribution to the community.

III. System Model

We present our system model in terms of a layered architecture following the proposed three layer architecture of Kramer and Magee [22] to address maintainability and separation of concerns principles. Additionally, we decided for this layered architecture to pay attention to different scopes and time horizons of the existing components. Further, we propose a feedback loop adaptation and discuss the relevant platooning coordination details.

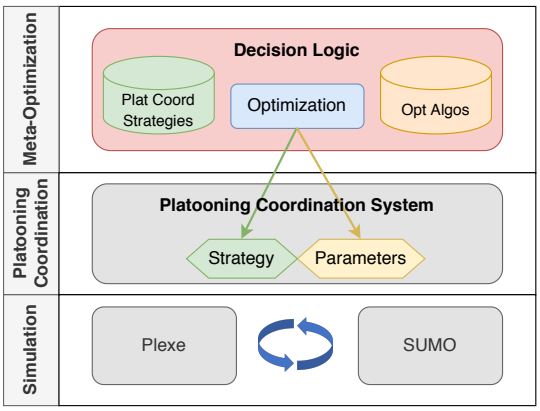

Fig. 1. Three-layered system model consisting of a simulation layer, a platooning coordination layer, and a layer for meta-optimization.

A. Layered Architecture

Figure 1 depicts our approach as a three-layer architecture. The *Simulation* forms the lowest layer of our architecture and provides the basis for vehicle traffic and platooning. This layer simulates aspects like traffic volumes, vehicle types, road conditions, and tempo limits using SUMO [23]. Plexe [24] and the Python Plexe API [25] provide the platooning functionality, that is, the execution of CREATE, JOIN, MERGE, SPLIT, and LEAVE tasks of vehicles. In this paper, we assume that a vehicle travels within the assigned platoon until it reaches its destination and we prohibit platoon assignment changes during the journey.

The second layer, called *Platooning Coordination*, contains a platooning coordination system and is responsible for executing the platooning coordination strategy and determining the platooning actions for each vehicle, that is, the decision which vehicle should CREATE, JOIN or LEAVE which platoon. A platooning coordination strategy makes this decision, which typically has several input parameters that can be tuned to optimize it for the current situation.

Finally, the *Meta-Optimization* layer contains a decision logic component. It incorporates a database of available platooning coordination strategies and optimization algorithms. For a specific platooning coordination strategy, the decision logic executes an optimization algorithm that tunes the according platooning coordination strategy's input parameters. After each iteration of the optimization algorithm, the platooning coordination strategy's tuned input parameters are passed to the Platooning Coordination layer and a simulation is executed to retrieve the resulting metrics for each set of input parameters.

B. Feedback Loop Adaptation

In addition to the already provided layered system model, we also present the process of our approach based on the

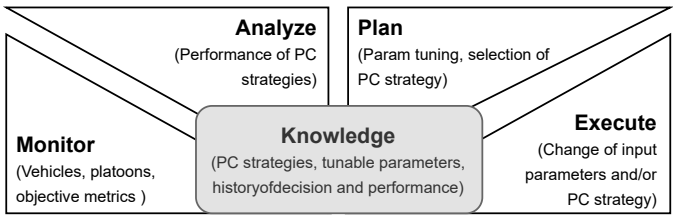

Fig. 2. Process of our approach depicted as MAPE-K Adaptation Loop as introduced in [26] (PC = platooning coordination).

MAPE-K adaptation loop [26] as depicted in Figure 2. In the *Monitor* phase, all use-case specific data is gathered, such as the number and properties of vehicles and platoons and their assignment. Further, all required details to calculate the optimization objectives—as explained later—are gathered and stored in the knowledge base. The *Analyze* phase uses this data to calculate the optimization objectives. These objectives are used to assess the performance of the current platooning coordination strategy. Based on this analysis, the *Plan* phase performs optimization tasks to tune the platooning coordination strategy's input parameters. Additionally, it is possible to change the platooning coordination strategy if another strategy better fits the current situation. Finally, the *Execute* phase uses these decisions and changes the platooning coordination strategy or its input parameters if required. The *Knowledge* component stores all relevant information of each loop, such as the monitored vehicle and platoon statistics, the analyzed performance, and the made decisions. This information can be used in future cycles to find better decisions on strategy selection and parameter tuning.

C. Use Case Details

Scenario Parameters. The *maximum platoon size* specifies the number of vehicles that are allowed to join a platoon. The *car spawn rate* defines the number of cars starting their route per hour while the *truck spawn rate* specifies the values for heavy-duty vehicles. The *platooning percentage* controls how many vehicles are participating in the platooning process. The parameters regarding the *platooning maneuvers overtaking and join* define whether platoons can overtake vehicles in a scenario and whether a vehicle can only join a platoon from the back or also at the front. Furthermore, the *number of lanes* defines the road's size. Finally, the *speed limit* affects the traffic flow by setting a maximum speed for every vehicle on the road. We select this set of scenario parameters since these form the most common parameter types for the strategies in the platooning coordination research.

Optimization Objectives. As target metrics for the optimization algorithms, we define four objectives with according metrics to analyze the currently selected input parameters' performance. All metrics are normalized to a range between zero and one. The *throughput* metric (ϕ) analyzes the number of vehicles arriving per hour compared to the number of vehicles starting per hour and covers platooning goals for traffic flow and road capacity:

$$\phi = \frac{\text{\# veh. arriving per hour}}{\text{\# veh. starting per hour}} \tag{1}$$

The *time loss* metric (δ) calculates the average amount of time in seconds all vehicles lost on the road compared to their expected travel time (tt). Therefore, we calculate the difference between actual and expected travel time, summarize this difference over all vehicles and divide it by the number of vehicles. Afterwards, this value is normalized by the average actual travel time of all vehicles. This metric is relevant for the velocity and time goals of platooning and vehicle-specific objectives for platooning coordination:

$$\delta = \frac{\text{avg. } tt - \sum(\text{actual } tt - \text{expected } tt)/\text{\# vehicles}}{\text{avg. } tt} \tag{2}$$

The *platoon utilization* metric (ρ) measures the average platoon size, that is the average number of vehicles in the platoons, compared to the maximum platoon size. Hence, it shows the mean utilization of the existing platoons with regards to the possible maximum platoon size. The aim of the optimization technique will be to maximize this value. This metric addresses platoon-specific goals of platooning coordination:

$$\rho = \frac{\text{avg. platoon size}}{\text{max. platoon size}} \tag{3}$$

Finally, the *platoon time* metric (τ) compares the time in seconds driven in a platoon to the overall travel time. This metric addresses the time and user comfort metrics of platooning:

$$\tau = \frac{\text{avg. time in platoon}}{\text{avg. travel time}} \tag{4}$$

IV. Platooning Coordination

This section introduces the platooning coordination strategies with input parameters and their impact on these strategies.

A. Strategies

The platooning coordination strategy is responsible for deciding which platoon a vehicle should join [27]. This decision is based on driver preferences like the desired velocity, the vehicle's current state, and the strategy's input parameters. While these strategies form state-of-the-art in platooning coordination and are applicable in many scenarios, we propose that they perform differently well in varying scenarios and an additional parameter optimization can tune their performance to a specific situation.

Best velocity. This strategy defines the best matching platoon by calculating the velocity difference between platoon and vehicle and selecting the platoon with the lowest positive speed delta. This can lead to matched platoons that are far away and we introduce two input parameters—a range for searching behind (*search dist. back*) and one for searching in front (*search dist. front*) of the vehicle.

Closest distance. The *closest distance* strategy analyzes the distance between vehicle and possible platoons and selects the platoon with the lowest longitudinal distance. This way,

vehicles can join close platoons very fast and thereby maximize their time in a platoon. We define the input parameter maximum speed difference that trades-off the differing desired velocity of the vehicle and planned velocity of the platoon.

Closest distance and lane. This strategy not only calculates the longitudinal distance of vehicle and platoon but incorporates the number of lanes between them. Therefore, this strategy uses the lane-distance ratio (ldr) as an additional input parameter. Equation 5 summarizes the calculation of the distance with regards to the longitudinal distance ($long\_dist$), the lane of the vehicle (v_lane) and of the platoon (p_lane).

$$dist = long\_dist + ldr * \mid v\_lane - p\_lane \mid \quad (5)$$

B. Input Parameters

Table I describes value ranges and default values for all input parameters. We set the value ranges for the search distance front and search distance back in meters to $[200, 1000]$ and $[0, 500]$, respectively. We define the default values as the mean values of the ranges: 600 and 250, respectively. The max speed difference parameter can take km/h values in the range of $[10, 60]$ and we set the default value to be 35. The lane-distance ratio parameter can accept values in the range of $[0, 500]$ and we define the default value to be 250. We further introduce parameters that apply to all strategies. The advertising duration describes the time (in seconds) a platoon stays on the lane it was created to remain joinable for all other vehicles. This parameter can accept values in the range of $[0, 20]$ and we set the default value for each strategy to be 10. The last set of parameters are the speed thresholds for different lanes. We specify speed thresholds for at most four lanes in this work, but transferring the concept to larger road segments is also possible. Based on each platoon's velocity, the platooning coordination strategy determines the starting lane for each platoon to create a pre-sorted road order. We define the ranges for the second, third, and the fourth lane to be $[100, 140]$, $[110, 150]$, and $[120, 160]$, respectively, and set the default values to be 100, 130, and 160, respectively. We decide to use these parameters to reserve the first lane for comparably slow truck vehicles driving below 100km/h and evenly distributed the speed ranges up to the maximum possible speed for platoons of 160km/h to the other lanes. If a platoon does not fit into the defined ranges per lane, a platoon spawns by default at the first lane and therefore, no range is given for it. Each input range is defined as integer, hence, assuming that no discretization is performed, the input configuration space is $800 * 500 * 50 * 500 * 20 * 40 * 40 * 40 = 1.28 * 10^{16}$.

TABLE I
VALUE RANGES AND DEFAULT VALUES FOR ALL INPUT PARAMETERS.

	Range	Default value
Search distance front (m)	[200, 1000]	600
Search distance back (m)	[0, 500]	250
Max speed difference (km/h)	[10, 60]	35
Lane-distance ratio (m)	[0, 500]	250
Advertising duration (s)	[0, 20]	10
Speed threshold lane 2 (km/h)	[100, 140]	100
Speed threshold lane 3 (km/h)	[110, 150]	130
Speed threshold lane 4 (km/h)	[120, 160]	160

V. Optimization Algorithms

We now describe the used optimization algorithms to adapt the coordination strategies by selecting their parameters.

A. NSGA-II

Since the non-dominated sorting genetic algorithm II [28] is a popular multi-objective optimization algorithm, we use the Python-based *inspyred* framework [29] due to the provided configuration possibilities. The tournament selection parameter specifies the number of solutions as parent elements for the next generation; we set this parameter to two. The algorithm uses the selected parents to generate the children by applying crossover and a mutation probability of 10%. As additional configuration parameters, the algorithm requires the population size and number of generations whose definition is postponed to the evaluation section.

B. Novelty Search

Inspired by Lehman et al. [30] we extend the NSGA-II algorithm by integrating novelty search. When evaluating the new solution's score, the added novelty function calculates the distance between the new solution and all solutions in the population and normalizes it into the novelty score (ns). This is used to calculate the output vector, i.e., the score value for each objective of all solutions, as proposed by Mouret et al. [31]. Equation 6 summarizes the calculation of the overall score. It adds the fitness value (f_i) of each objective i that is weighted using the novelty weight (w) to the weighted novelty score (ns). We examine the definition of the novelty weight (w) value experimentally throughout our evaluation.

$$s_i = f_i * (1 - w) + ns * w \quad (6)$$

The result of this process is a score vector for each solution containing a score for each objective. Afterward, the algorithm updates the novelty archive inspired by Fredericks et al. [7]. We use a fixed maximum size of 20% of the total evaluations done by the optimization algorithm. The algorithm adds all solutions of the new generation to the novelty archive, sorts it by the novelty score of the solutions, and cuts off the worst solutions until the archive's maximum size is met. This novelty archive is then used for the next generation of solutions.

C. Bayesian Optimization

Bayesian Optimization is a popular single-objective optimization algorithm implemented in the *scikit-optimize* framework for Python [32] which we use in this work. We extend it to handle multi-objective problem statements by implementing a hypervolume [33] and a Pareto dominance [34] approach to merge the multi-objective score function into one single value. The hypervolume approach uses a reference point with

a higher value than any objective value and compares the solutions to this reference point. Since all objective values are normalized to a range from zero to one, we specify the value 1.1 for every objective as the reference point. We decided to use this value as it is outside of the normalization range of the objective values. Still, we could have chosen any value above 1.0 as this has no impact on the optimization. Afterward, the algorithm uses the hypervolume implementation by Simon Wessing [35] and the given reference point to calculate the score of the new solution. The Pareto dominance approach rates the new solution based on the percentage of solutions of the current Pareto front not dominating it. Utilizing these black-box function, the Bayesian optimization procedure builds its model based on a single value and tries to find better solutions sequentially. The algorithm needs configuration parameters for the number of random initialization points and the number of evaluations of the black-box function, which are experimentally examined in the evaluation.

D. Simulated Annealing

As fourth optimization algorithm, we use Simulated Annealing [36] in this work. Similar to Bayesian Optimization, Simulated Annealing supports only one return value from the score function as well. Therefore, we again apply the hypervolume and Pareto dominance approaches as already explained for the Bayesian Optimization in Section V-C to address multi-objectiveness. Simulated Annealing requires the definition of some parameters. We set the temperature reduction factor to 0.5, which leads to halving the temperature after each iteration. Further, the initial temperature is determined at runtime by calculating four randomly chosen solutions' mean score. According to Rao [36], the number of iterations should be set within a range of 50 to 100 iterations. This value and the maximum number of evaluations are again examined experimentally in the evaluation section.

VI. Evaluation Testbed

This section introduces our third contribution, the design of a customizable and reusable testbed for evaluating meta-optimized adaptation planning algorithms. We propose this testbed on a conceptual level than can be applied to a diverse set of meta-optimized adaptation planning algorithms. However, since the main components, that are the Controller, Simulation, Optimization, and Analysis heavily depend on the use case we do not provide an open source software framework. Figure 3 summarizes our evaluation approach.

Controller: The controller first defines the scenario parameters and the strategy to be used and forwards this information to the *Simulation*. We define four base scenarios representing general traffic situations. Table III describes our scenarios. In the following, we describe the different parameters. Platooning percentage captures the number of vehicles that are interested in platooning. We defined the vehicle spawn rates using real traffic information—provided by the Federal Highway Research Institute [37] —from the simulated road segment of the German motorway A8, ranging from the motorway

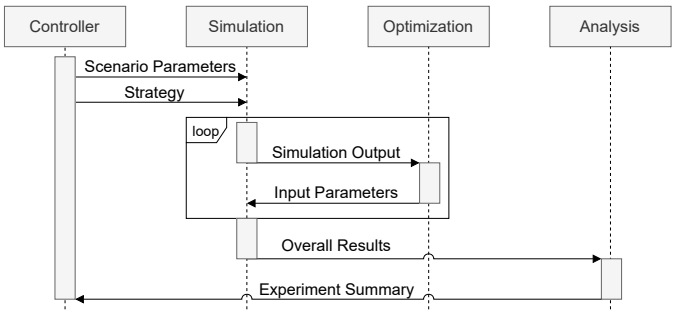

Fig. 3. Sequence diagram of the evaluation procedure handled by a Controller that manages the Simulation, Optimization, and Analysis components.

intersection Stuttgart until the exit ramp Stuttgart-Degerloch. In 2018, on average 3351 cars and 434 trucks drove on this road segment per hour; Scenarios 2 and 3 select values for vehicle spawn rates above the average (but still below the maximum traffic volume), Scenarios 1 and 4 utilize low traffic volumes instead. The simulation environment handles vehicle spawns with the same random seed during all the simulation runs. Following German traffic laws, we consider two different speed limits for platoons: the speed limit of 160 km/h creates traffic scenarios with few restrictions for velocity; alternatively, vehicles can only drive 120 km/h at maximum. We selected a maximum speed limit of 160 as high speed platooning is not realistic. This is due to the effect that the faster a platoon travels, the more difficult it is to ensure string stability. Further, our experience showed that vehicles inside the simulation exceed this limit very seldom even if they are allowed and able to drive faster. Hence, this speed limit will affect a small amount of vehicles only and can be considered as driving without restrictions. Further, we modified the parameters maximum platoon size (5 versus 10 vehicles), platoons can overtake other traffic participants or not, and—as the simulated highway supports a temporary release of the hard shoulder—the number of lanes (3 versus 4). A limit for the number of vehicles in a platoon is reasonable since the disturbance propagation within the platoon increases with the platoon size [38]. Table II lists these variations. As a result, the alteration of the basic scenarios with the five variations of the mentioned parameters generates twenty scenarios for the evaluation.

TABLE II
Used variations for the base scenarios.

	Max. platoon size	Overtaking	Number of lanes
Default values	5	yes	4
Variation 1	5	yes	3
Variation 2	10	yes	4
Variation 3	5	no	4
Variation 4	10	no	3

Simulation: We use SUMO (Simulation of Urban MObility) [23] for simulating traffic with the extension Plexe [24]

which provides platooning functionality. To coordinate the platooning operation, we integrate the Platooning Coordination System (PCS) from [27]. Using the *Python-Plexe* API [25], the PCS periodically receives information for each vehicle, such as current position and velocity. In return, using the currently specified platooning coordination strategy, the PCS sends commands—for example, changing velocity or the lane—for each vehicle to the Python-Plexe for controlling platooning.

Optimization: The optimization step adapts the parameters of the platooning coordination strategy to the current traffic situation. The used optimization algorithm is interchangeable but needs to optimize a black-box multi-objective function without prior knowledge. We apply NSGA-II [28], Novelty Search [39], Bayesian optimization [40], and Simulated Annealing [41] as optimization techniques (see Section V). All of these approaches support multiple objectives based on Pareto fronts [34], the concept of hypervolumes [33], or the dominance rank approach [42]. The input for the black-box function is a possible configuration of the platooning coordination strategy. A simulation run return metrics representing the fitness of the function evaluation. The optimization algorithm alternately evaluates configurations for the function and computes new configurations. This iterative cycle repeats itself until it reaches a predefined maximum number of function evaluations.

TABLE III
THE PARAMETER SETTINGS FOR THE FOUR BASE SCENARIOS.

	Scen. 1	Scen. 2	Scen. 3	Scen. 4
Platooning percentage	25%	50%	75%	100%
Car spawn rate (car/h)	2680	4021	4021	2680
Truck spawn rate (truck/h)	349	524	524	349
Speed limit (km/h)	120	160	120	160

Analysis: Finally, the Analysis receives log data from the *Simulation* and *Optimization* loop. It analyzes the optimized strategy's performance by preprocessing the log data and calculating the platooning and performance metrics.

VII. EVALUATION

The purpose of the evaluation is two-fold. First, Section VII-A describes experiments to motivate (i) situation-awareness triggering the need to switch the adaptation planning strategy in the platooning coordination use case as well as (ii) that different configurations of the same strategy are beneficial depending on the current situation of the environment and the system itself. In our use case, the range of the parameters platooning percentage (in percent), number of cars and trucks, speed limit (80, 90, 100, 110, 120, 130, unlimited), and number of lanes lead to $100*10,000*1,500*6*5 = 4.5*10^{10}$ different traffic scenarios, i.e., environment situations. As it will not be possible to identify the best strategy and its optimal configuration for each specific situation, we claim that it is beneficial to have a mapping of some everyday situations to adaptation planning strategies and then optimize the configuration of the strategy. Accordingly, Section VII-B shows that the system should not only switch the adaptation planning strategy according to the current situation but also optimize its parameters. Further, we provide an analysis of different optimization procedures.

A. Comparison of Platooning Coordination Strategies

For supporting our claim of situation-awareness in the behavior of platooning coordination strategies, we focus in the first experiments on showing that the choice of the platooning coordination strategies itself is dependent on the current situation. Hence, we omit optimizing a strategy's parameters and use the default configurations and two variants of each of the platooning coordination strategies from Section IV-A for evaluation. We apply those strategies in all of the 20 traffic scenarios. Table IV displays the changes to each default configuration that lead to the respective configuration sets.

TABLE IV
CHANGES TO THE DEFAULT CONFIGURATION PARAMETERS OF THE PLATOONING COORDINATION STRATEGIES.

Strategy	Configuration	Parameters changing	Default	New
Best velocity	Configuration 1	None (default)	-	-
	Configuration 2	Speed threshold lane 2	100	110
		Speed threshold lane 3	130	125
		Speed threshold lane 4	160	140
	Configuration 3	Advertising duration	10	5
		Search distance front	600	400
		Search distance back	250	200
Closest distance	Configuration 1	None (default)	-	-
	Configuration 2	Speed threshold lane 2	100	110
		Speed threshold lane 3	130	125
		Speed threshold lane 4	160	140
	Configuration 3	Advertising duration	10	5
		Max speed difference	35	20
Closest distance and lane	Configuration 1	None (default)	-	-
	Configuration 2	Speed threshold lane 2	100	110
		Speed threshold lane 3	130	125
		Speed threshold lane 4	160	140
	Configuration 3	Advertising duration	10	5
		Max speed difference	35	20
		Lane-distance-ratio	250	150

1) Situation-Dependent Behavior of the Coordination Strategies: In this experiment, we compare the strategies in all 20 traffic scenarios and analyze, which strategy optimizes which metric. Figure 4 displays the comparison between the strategies. The figure considers the four metrics independently. The x-axis displays the different platooning coordination strategies; the y-axis shows the number of best solutions each strategy has for each metric. Since 20 different traffic situations exist, a coordination strategy can have 20 best solutions at maximum. As every platooning coordination strategy has three different configuration sets for this run, a coordination strategy may score the best solution with any of those configurations. To have the best solution for a given metric and traffic situation, a coordination strategy needs the best objective score for the metric and scenario variation. As can be seen, the best velocity strategy performs best, i.e., has the highest number of best solutions, for the platoon time and platoon

utilization metric what goes in line with our expectations as the selected platoon fits best regarding the travel speed and the need to switch to another platoon is minimized in this strategy. Contrary, the closest distance as well as the closest distance and lane strategies perform best with regards to the time loss and throughput metric. This might be due to the fact, that the closest distance strategy minimizes the time until a vehicle actually joined a platoon and, hence, avoids long catch up times with a platoon ahead. Thus, none of the strategies performs best for each metric. Consequently, the relevant metrics drive the choice of the platooning coordination strategy and, hence, this choice is objective-dependent. Further, as, e.g., each strategy optimizes the *throughput* metric in a specific scenario, even for a dedicated metric, a specific strategy might be superior in a specific traffic scenario; hence, the choice is also situation-dependent. Hence, the results show that adapting to the traffic situation by switching strategies is beneficial.

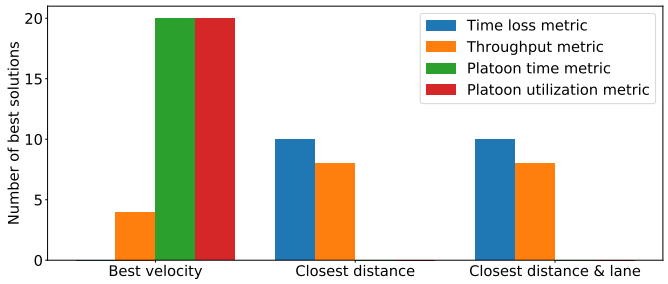

Fig. 4. Comparison of the coordination strategies over all 20 traffic situations.

2) Situation-Dependent Behavior of the Strategy Configuration: We now focus on one specific strategy and analyze in detail the relevance to take various configurations into account. The experiment focuses on the metric *platoon utilization*. For this metric, the strategy *best velocity* has the best solutions in all of the 20 traffic scenarios. However, each of the three different strategy configurations might be superior depending on the traffic condition. Figure 5 shows for every scenario how many best solutions each configuration of the *best velocity* strategy has. Since every scenario has the default setting and four additional variations, the maximum number of best solutions per scenario is five. The figure shows that Configuration 2 only has the best solutions in Scenario 2 and 3. The other configurations have the best solutions in every scenario. Furthermore, there is no scenario with only one configuration having the best *platoon utilization* solution for every variation. Consequently, the results demonstrate the benefit of adapting the platooning coordination strategy's configuration to the specifics of the current traffic situations as even slight changes in the conditions need to be coped in the strategy's parameters.

B. Optimization of the Strategy Parameters

The previous experiments motivate that optimizing the platooning coordination strategies' configuration parameters based on the traffic scenario may be beneficial. Table V summarizes the optimization algorithms—including the configuration variants of every algorithm—used in the experiments. Every algorithm has 1000 function evaluations before terminating with the final Pareto front. For the evolutionary approaches, the 1000 evaluations are split into 50 generations with 20 population each. Since Novelty Search is less explored than the other optimization techniques, this evaluation run considers three variants of Novelty Search with 10%, 20%, and 30% novelty weight. Bayes and Simulated Annealing require an extension for supporting multi-objectiveness. Therefore, the evaluation includes two variants of these optimization algorithms. One variant using the hypervolume value and one utilizing the dominance rank approach. We consider only the default variant of traffic scenario 2 and apply the *best velocity*, *closest distance*, and *closest distance & lane* strategies.

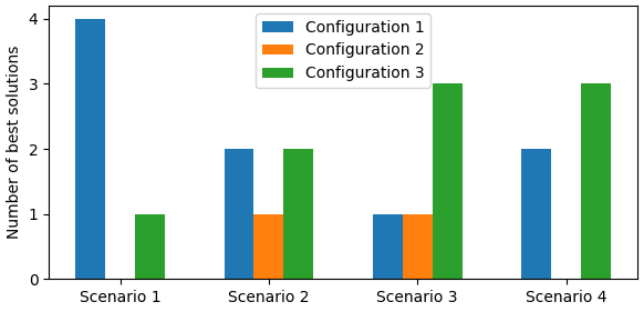

Fig. 5. Analyzing the velocity based strategy: Best configuration parameters dependent on the scenario for the platoon utilization metric.

TABLE V
THE OPTIMIZATION ALGORITHMS USED IN THE EXPERIMENTS (NOVSEA = NOVELTY SEARCH; SIMAN = SIMULATED ANNEALING; POP = POPULATION; EVAL = EVALUATIONS).

Algorithm	Multi-obj.	Configuration params
NSGA-II	Pareto front	Pop. size: 20 #gen: 50
NovSea 0.1	Pareto front	Pop. size: 20 #gen: 50 Nov weight: 0.1
NovSea 0.2	Pareto front	Pop. size: 20 #gen: 50 Nov weight: 0.2
NovSea 0.3	Pareto front	Pop. size: 20 #gen: 50 Nov weight: 0.3
Bayes hv	Hypervolume	#init points: 20 #eval.: 1000
Bayes dom	Dom. count	#init points: 20 #eval.: 1000
SimAn hv	Hypervolume	#iterations: 100 #eval.: 1000
SimAn dom	Dom. count	#iterations: 100 #eval.: 1000

1) Comparing the Course of Optimization: This experiment compares all eight optimization approaches based on the course of their optimization, focusing on the progress in solution quality and the change in the Pareto front size during the optimization procedure. Since the duration of the optimization process heavily depends on the executing hardware, we decided—following the common evaluation of optimization techniques—to use the number of function evaluations as duration metric. Figures 6 and 7 show important aspects for this evaluation. Figure 6 shows the progress in solution quality. The x-axis displays the number of performed function evaluations.

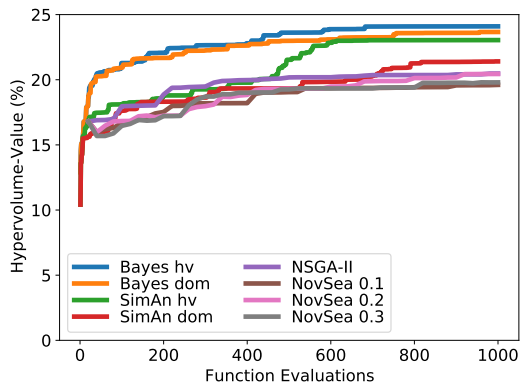

Fig. 6. Pareto front quality in the course of optimization.

For analyzing the quality of the optimization at a given point in time, this plot takes the Pareto front at this timestamp, calculates the hypervolume value for this front, and shows the percentage of the maximum volume this Pareto front covers on the y-axis. The optimization algorithms optimize four objectives at once, all normalized between zero—for the worst possible score—and one. In this setting, the hypervolume values become small very fast. For example, a solution scoring 0.8 in every objective score only has a hypervolume percentage of $0.8^4 = 0.4096 = 40.96\%$. Since the evaluation procedure covers three different platooning coordination strategies, the hypervolume value for every optimization algorithm and function evaluation is an average value of all coordination strategies. We decided to use the Hypervolume value since we are interested in convergence and diversity aspects of a Pareto front [43]. However, other quality indicators for comparing Pareto fronts exist that could be used when focusing other aspects [44]. Figure 7 shows the size of the Pareto front over the course of optimization.

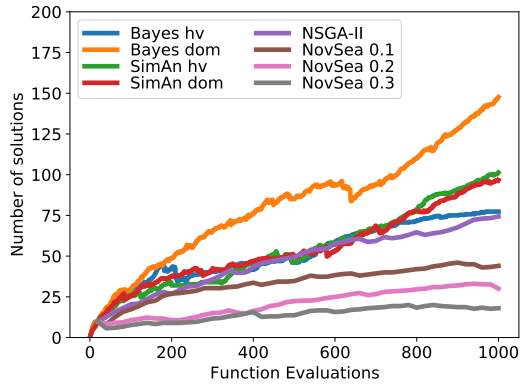

Fig. 7. Pareto front size in the course of optimization.

Similar to the first plot, the x-axis displays the number of function evaluations. The y-axis shows the number of solutions in the front at any given point during optimization. This value considers all coordination strategies by taking the average Pareto front size of all three. The results show Bayes improving its hypervolume the fastest. The Bayesian approaches and the hypervolume variant of Simulated Annealing are pretty close for the best hypervolume score regarding the final solution. However, they might have an advantage for the hypervolume metric because they directly improve this metric as the goal of their optimization. Out of the evolutionary optimization techniques, NSGA-II improves the fastest at first, in terms of the final score, NSGA-II and Novelty Search 0.2 perform equally. However, the other Novelty Search versions have a lower score. After around 600 function evaluations, the hypervolume progress stagnates for all optimization techniques except for Simulated Annealing. The size of the Pareto front still rises from 600 function evaluations onwards, though. This indicates that the optimization algorithms only find a small number of new solutions dominating the existing ones because replacing a solution with a dominating one generally comes with a bigger increase in the hypervolume value. NSGA-II produces the highest number of solutions for the evolutionary approaches, followed by Novelty Search 0.1. Novelty Search 0.3 has the lowest size. This shows how it becomes more difficult for Novelty Search with increasing novelty weight as new solutions become less novel and thus have fewer chances to be added to the front. This scales with the novelty weight. Bayes with dominance rank evaluation has the highest number of solutions, Simulated Annealing and the other Bayes variant are comparable to NSGA-II in terms of Pareto front size. We are aware, that the currently discussed results have limited expressive power when applying the approach in on-line optimization of adaptation strategies scenarios since we did not perform time measurements of the optimization process. However, we applied anytime optimization approaches—such as Simulated Annealing, NSGA-II, and Novelty Search—that yield valid results after each function evaluation and thus a final convergence of the algorithms is not mandatory for executing optimized adaptations. Further, in our application scenario, the required time for on-the-fly adaptation of the parameters is less important, as we only adjust the strategy's parameters after switching the planning strategy for the current situation, i.e., we do not have real-time constraints. From similar applications of the optimization algorithms it seems valid to assume a reasonable time. Still, for a real-world application of our approach, time measurements are meaningful to select the best fitting optimization technique.

2) Comparing the Final Pareto Fronts: This experiment analyzes the solutions returned by the optimization algorithms after creating the final Pareto front.

Evaluating the Algorithms based on Pareto Dominance: First, we compare the optimization algorithms using only the concept of Pareto dominance. Figure 8 combines the solutions of all Pareto fronts produced by every optimization algorithm into one front. First, it displays the size of the final Pareto fronts in green. Additionally, the plot contains the number of contributions and unique contributions for every optimization algorithm. The contribution indicates the number of solutions in the final front produced by an optimization algorithm which are not dominated by solutions in the combined front. A unique contribution is a contribution that was only made by

TABLE VI
AVERAGE OBJECTIVE SCORE AND STANDARD DEVIATION FOR EVERY OPTIMIZATION ALGORITHM AND OBJECTIVE.

	Time loss		Throughput		Platoon time		Platoon util.	
	mean	std	mean	std	mean	std	mean	std
Bayes hv	0.887	0.015	0.732	0.007	**0.475**	0.159	**0.572**	0.115
Bayes dom	**0.907**	0.016	**0.737**	0.005	0.402	0.175	0.486	0.128
SimAn hv	0.89	0.014	0.733	0.007	0.461	0.142	0.559	0.087
SimAn dom	0.902	0.013	0.736	0.006	0.4	0.14	0.505	0.104
NSGA-II	0.905	0.015	**0.737**	0.006	0.39	0.137	0.502	0.124
NovSea 0.1	0.9	0.014	0.734	0.006	0.379	0.132	0.5	0.114
NovSea 0.2	0.897	0.014	0.734	0.006	0.396	0.143	0.517	0.121
NovSea 0.3	0.893	0.013	0.732	0.006	0.405	0.127	0.526	0.105
Default	0.897	0.009	0.735	0.003	0.363	0.033	0.494	0.034

one optimization algorithm. Ranking the optimization techniques, Bayes is first in terms of contributions, followed by Simulated Annealing and NSGA-II, which provide comparable results, and finally Novelty Search with low amounts of contributions. However, every optimization technique—except for Novelty Search with a novelty weight of 0.3—contributes some unique solutions to the combined final Pareto front. Thus, no algorithm dominates another one entirely.

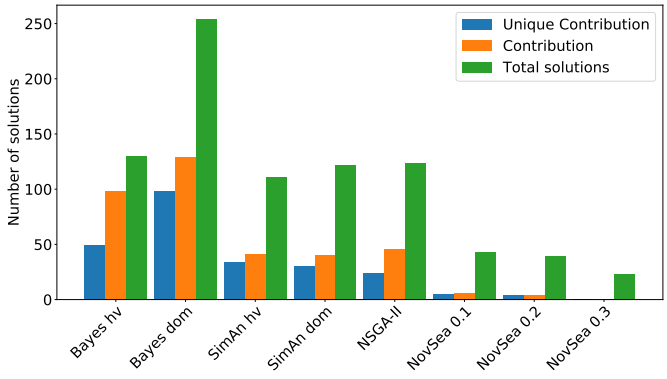

Fig. 8. Contribution to the best-known front by each optimization algorithm.

Evaluating the Algorithms based on Objective Scores: Second, we analyze the final Pareto fronts of every optimization algorithm based on the objective scores as summarized in Table VI. Remember, we only use Scenario 2 (c.f. Table III) for this evaluation, apply the three strategies, and calculate mean and standard deviations over the final Pareto fronts of all strategies per optimization technique. The table highlights the best average value per objective. Taking all platooning coordination strategies into account, the evaluation calculates the average and standard deviation over the final fronts of all strategies. Overall, Bayes has great results for every metric, especially when taking both algorithm versions into account. NSGA-II and Simulated Annealing show average values overall. Novelty Search is below average, except for the *platoon utilization* metric of versions 0.2 and 0.3 as well as the *time loss* metric for Novelty Search 0.1. Furthermore, there is a tendency for the hypervolume and dominance rank versions of Simulated Annealing and Bayes. The solutions produced by the dominance rank show high scores for *time loss* and *throughput* while the hypervolume versions dominate the platooning-specific metrics. As mentioned previously, the hypervolume based approaches may focus on more fluctuating metrics since a greater increase in metric score leads to better hypervolume values. This complies with the findings in this table, showing a higher standard deviation in the platoon metrics. Additionally, the high standard deviation of Bayes with dominance rank is another interesting aspect. Figure 8 shows a high Pareto front size of Bayes with dominance rank combined with higher standard deviation indicate that this optimization algorithm finds more diverse solutions providing opportunity for adaption to user preferences.

C. Discussing the relation between optimization, adaptation planning, and objectives

The following Figure 9 contains boxplots for showing the relation between optimization algorithms, platooning coordination strategies, as well as objectives. As it can be seen, to optimize different objectives, not only a specific planning strategy is superior, also for the optimization different algorithms might improve the planning strategies differently. As one specific example, for the objective platooning utilization, it can be seen that for the strategy *best velocity* NSGA-II performs best. However, it performs worst for the *closest distance*; contrary behaves Simulated Annealing. Also, it can be seen that for a specific planning strategy, not a single optimization algorithm performs best. For example, for the *closest distance* and lane strategy, the Simulated Annealing algorithm performs worst when targeting the objective *time loss*. However, it is superior for the *platoon utilization* objective. Finally, it can be concluded that the choice of the adaptation planning strategy but also the optimizer to improve the strategy's parameters is not a "one fitting all" choice, especially in multi-objective scenarios. Still, it can be visible that in most scenarios, the application of any of the three considered optimization techniques still outperforms the default parameters of the adaptation planning strategies' default parameters, which has been already defined by domain experts.

D. Threats to Validity

We have identified the following threats to validity of the evaluation results. First, we only applied a small set of adaptation planning strategies for platooning coordination, while additional strategies are present in literature. However, as our focus was not on finding the best solution for the domain but showing the requirements to switch the adaptation planning strategy, our choice of strategies should be representative enough. Second, we have used the default version of the optimization strategies with only slightly adjusted variants. Thus, we did not tune the meta-parameters (e.g., number of generations, crossover rate) to tailor each technique specifically to the use case. Third, we focus on platooning coordination as use case and our evaluation results are obtained for this simulated system. However, we assume that the results are

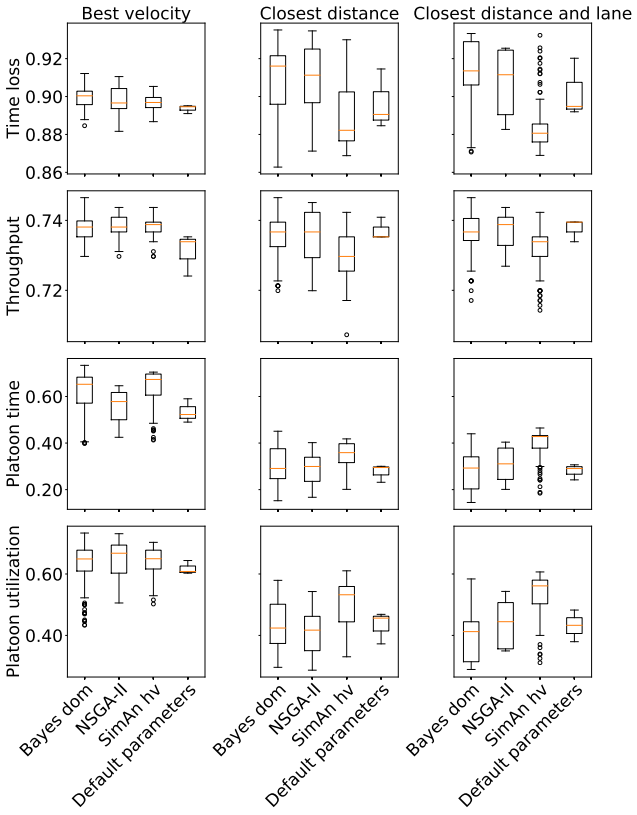

Fig. 9. Comparison of optimized and default solutions for the strategies.

representative for many other classes of self-adaptive systems. Fourth, we examined the simulations in a single map and, due to high computational costs of the optimization, with a restricted number of twenty situations. A more in-depth study could examine multiple maps and additional situations. Lastly, it is not preferable to perform those optimizations at runtime in a real system as it might lead to performance decreases and uncertainties in the objective space. Hence, a simulation of the system is required to identify the best strategy and its parameter setting for a specific situation.

VIII. Challenges

We now discuss challenges for our vision of a situation-aware meta-optimization framework in line with [7].

Flexible situation detection. In this paper, we excluded the definition of a specific situation detection approach and solely consider the detection of a situation *a posteriori* and perform a reactive analysis. Still, our system model supports on-the-fly situation detection. In line with current research efforts in the area of SASs [9], [45], the integration of predicting system states seems promising to enable proactive adaptation.However, those approaches might introduce additional uncertainty as predictions of situations always come with some unreliability. In a previous study, we analyzed how to cope with uncertainty in the domain of Intelligent Transportation Systems [46]. In the future, further studies need to be conducted to analyze the scalability and transferability of the situation detection and the system as a whole.

Domain knowledge integration. Currently, domain knowledge is applied to set up the adaptation planning strategies' default parameter settings. However, the integration of domain knowledge might be beneficial for finding the best adaptation planning strategy as well as for boosting the optimization process by reducing the search spaces. The main challenge is the transformation of domain-specific descriptions from users that potentially are neither familiar with self-adaptive system nor with programming in a domain-independent form that can be understood by the framework. A further advantage of domain knowledge integration might be the search space reduction for the optimization component and, hence, a faster time to result in this component.

Meta-adaptation. One crucial challenge is the meta-adaptation, i.e., the re-configuration of the adaptation planning strategy at runtime. In [47], we compared different approaches for structural and parametrical meta-adaptation of self-adaptive systems. Structural meta-adaptation changes the structure of the decision logic, e.g., the exchange of the adaptation planning strategy. Parametrical meta-adaptation refers to the adjustment of parameters, e.g., changing the configuration of the adaptation planning strategy. Our approach requires addressing both structural and parametrical meta-adaptation as it exchanges the adaptation planning strategy and customizes its parameters to the specifics of the current situation. None of the analyzed approaches from [47] provide such flexibility, especially the interplay of strategy selection and simultaneously optimizing its parameters might be challenging.

IX. Conclusion

In this paper, we make the first step towards our vision of a framework for situation-aware meta-optimization of adaptation planning strategies by our threefold contribution: (i) we propose a layered system model for the situation-aware selection of adaptation planning strategies, (ii) an adapted feedback loop for meta-optimization of the current strategy's parameters to the system / environmental situation, and (iii) propose a customizable and reusable testbed for evaluating meta-optimized adaptation planning algorithms. We provide a study within the platooning coordination domain and show situation- and objective-dependent behavior of the coordination strategies and their configuration. In our opinion, platooning coordination suits as domain under study as diverse so-called platooning coordination strategies exist in the literature; however, according to the no-free-lunch theorem, each has individual assets and drawbacks, making them best applicable for different traffic situations. Additionally, we compare eight different versions of four optimization algorithms to tune the coordination strategies' parameters to provide advice for selecting the optimization algorithm for our meta-optimization framework. In the future, we plan to tackle the mentioned challenges and focus on implementing the meta-optimization framework.This includes the implementation of a flexible situation detection, e.g., based on clustering (cf. [7]) or classification (cf. [48]) techniques. Further, the integration of domain knowledge can improve the process.However, this

might result in issues for human in the loop integration, e.g., uncertainty [49]. Finally, future work could integrate approaches reducing the search space such as [50], [51].

References

[1] "Wissenswertes zur Verkehrsstatistik für die Jahre 2018/2019," https://www.bussgeldkatalog.org/verkehrsstatistik/#auch_in_puncto_stau_zeigt_die_statistik_eine_steigerung_im_jahr_2018, access: 2020-02-20.

[2] L. Figueiredo *et al.*, "Towards the development of intelligent transportation systems," in *Proceedings of the IEEE Intelligent Transportation Systems. (Cat. No. 01TH8585)*. IEEE, 2001.

[3] T. Robinson, E. Chan, and E. Coelingh, "Operating platoons on public motorways: An introduction to the sartre platooning programme," in *17th World Congress on Intelligent Transport Systems*, vol. 1, 2010, p. 12.

[4] A. Alam, "Fuel-efficient distributed control for heavy duty vehicle platooning," Ph.D. dissertation, KTH Stockholm, 2011.

[5] T. Sturm *et al.*, "A taxonomy of optimization factors for platooning," *IEEE Transactions on ITS*, pp. 1–18, 2020.

[6] V. Lesch *et al.*, "A Comparison of Mechanisms for Compensating Negative Impacts of System Integration," *Future Generation Computer Systems*, vol. 116, 2021.

[7] E. M. Fredericks *et al.*, "Planning as Optimization: Dynamically Discovering Optimal Configurations for Runtime Situations," in *SASO*. IEEE, 2019, pp. 1–10.

[8] B. H. C. Cheng *et al.*, *Software Engineering for Self-Adaptive Systems: A Research Roadmap*. Springer Berlin Heidelberg, 2009.

[9] C. Krupitzer *et al.*, "A Survey on Engineering Approaches for Self-Adaptive Systems," *PMCJ*, vol. 17, no. Part B, pp. 184–206, 2015.

[10] ——, "Adding Self-Improvement to an Autonomic Traffic Management System," in *Proc. ICAC*, 2017.

[11] C. Neumüller *et al.*, "Large scale parameter meta-optimization of metaheuristic optimization algorithms with heuristiclab hive," *Actas del VIII Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados*, 2012.

[12] M. Feurer, J. T. Springenberg, and F. Hutter, "Initializing bayesian hyperparameter optimization via meta-learning," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[13] R. Chis, M. Vintan, and L. Vintan, "Multi-objective dse algorithms' evaluations on processor optimization," in *2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2013, pp. 27–33.

[14] L. Vinţan *et al.*, "Improving computing systems automatic multiobjective optimization through meta-optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 7, pp. 1125–1129, 2015.

[15] P. Lewis *et al.*, "Towards a framework for the levels and aspects of self-aware computing systems," in *Self-Aware Computing Systems*. Springer, 2017, pp. 51–85.

[16] I. Gerostathopoulos *et al.*, "Strengthening adaptation in cyber-physical systems via meta-adaptation strategies," *ACM Transactions on Cyber-Physical Systems*, vol. 1, no. 3, pp. 1–25, 2017.

[17] C. Kinneer *et al.*, "Managing uncertainty in self-adaptive systems with plan reuse and stochastic search," in *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 40–50.

[18] R. Calinescu *et al.*, "Understanding uncertainty in self-adaptive systems," in *ACSOS*. IEEE, 2020, pp. 242–251.

[19] T. Hardes and C. Sommer, "Towards heterogeneous communication strategies for urban platooning at intersections," in *2019 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2019, pp. 1–8.

[20] B. Porter and R. Rodrigues Filho, "Losing control: The case for emergent software systems using autonomous assembly, perception, and learning," in *SASO*. IEEE, 2016, pp. 40–49.

[21] S. Kang, T. Choi, and T. P. Pavlic, "How far should i watch? quantifying the effect of various observational capabilities on long-range situational awareness in multi-robot teams," in *ACSOS*. IEEE, 2020, pp. 146–152.

[22] J. Kramer and J. Magee, "Self-managed systems: an architectural challenge," in *Future of Software Engineering*. IEEE, 2007.

[23] M. Behrisch *et al.*, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.

[24] M. Segata *et al.*, "PLEXE: A Platooning Extension for Veins," in *6th IEEE Vehicular Networking Conference*. IEEE, 2014.

[25] "Plexe APIs for python," https://github.com/michele-segata/plexe-pyapi, access: 2020-02-19.

[26] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[27] C. Krupitzer *et al.*, "A Modular Simulation Framework for Analyzing Platooning Coordination," in *Proceedings of the 1st ACM Workshop on TOP-Cars, Colocated with ACM MobiHoc 2019*. ACM, July 2019.

[28] K. Deb *et al.*, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[29] "inspyred: Bio-inspired Algorithms in Python," https://pythonhosted.org/inspyred/, access: 2020-06-12.

[30] J. Lehman and K. O. Stanley, "Efficiently evolving programs through the search for novelty," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 2010, pp. 837–844.

[31] J.-B. Mouret, "Novelty-based multiobjectivization," in *New Horizons in Evolutionary Robotics*. Springer, 2011, pp. 139–154.

[32] "scikit-optimize: Sequential model-based optimization in Python," https://scikit-optimize.github.io/stable/index.html, access: 2020-06-12.

[33] C. M. Fonseca, L. Paquete, and M. López-Ibánez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *2006 International Conference on Evolutionary Computation*. IEEE, 2006.

[34] D. A. Van Veldhuizen and G. B. Lamont, "Evolutionary computation and convergence to a pareto front," in *Late-Breaking Papers at the Genetic Programming 1998 Conference*. Citeseer, 1998, pp. 221–228.

[35] "Hypervolume - Technische Universität Dortmund," https://ls11-www.cs.tu-dortmund.de/rudolph/hypervolume/start, access: 2020-06-09.

[36] S. S. Rao, *Engineering optimization: theory and practice*. John Wiley & Sons, 2019.

[37] "bast - Bundesanstalt für Straßenwesen," https://www.bast.de/BASt_2017/DE/Home/home_node.html.

[38] J. Eyre, D. Yanakiev, and I. Kanellakopoulos, "A simplified framework for string stability analysis of automated vehicles," *Vehicle System Dynamics*, vol. 30, no. 5, pp. 375–405, 1998.

[39] J. Lehman and K. O. Stanley, "Exploiting open-endedness to solve problems through the search for novelty." in *ALIFE*, 2008, pp. 329–336.

[40] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[41] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[42] G. F. Smits and M. Kotanchek, "Pareto-front exploitation in symbolic regression," in *Genetic Programming Theory and Practice II*. Springer, 2005, pp. 283–299.

[43] S. Wang *et al.*, "A practical guide to select quality indicators for assessing pareto-based search algorithms in search-based software engineering," in *Proceedings of the 38th International Conference on Software Engineering*, 2016, pp. 631–642.

[44] M. Li and X. Yao, "Quality evaluation of solution sets in multiobjective optimisation: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–38, 2019.

[45] D. Weyns, "Software engineering of self-adaptive systems," in *Handbook of Software Engineering*. Springer, 2019, pp. 399–443.

[46] V. Lesch *et al.*, "Utility-based vehicle routing integrating user preferences," in *Proceedings of International Workshop on Pervasive Computing for Vehicular Systems*. IEEE, March 2021.

[47] C. Krupitzer *et al.*, "Comparison of Approaches for Self-Improvement in Self-Adaptive Systems," in *Proc. ICAC*, 2016, pp. 308–314.

[48] ——, "Hips do lie! a position-aware mobile fall detection system," in *Proc. PerCom*, 2018.

[49] S. Mahdavi Hezavehi, P. Avgeriou, and D. Weyns, *A Classification Framework of Uncertainty in Architecture-Based Self-Adaptive Systems With Multiple Quality Requirements*, 01 2017, pp. 45–77.

[50] D. Pukhkaiev and S. Götz, "Brise: energy-efficient benchmark reduction," in *Proceedings of the 6th International Workshop on Green and Sustainable Software*, 2018, pp. 23–30.

[51] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization*, C. A. C. Coello, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 507–523.