Engine Activation Planning for Series Hybrid Electric Vehicles

Kyle Hollins Wray, Richard Lui, and Liam Pedersen

***Abstract*— We present a solution for intelligent planning of engine activations for series hybrid electric vehicles (HEVs). Beyond minimizing energy expenditure, other real-world objectives must be incorporated, such as minimizing the perceived engine noise and the frequency of mode transitions between activation and deactivation. We model this problem as a multi-objective stochastic shortest path (MOSSP) problem that takes a vehicle model and navigation map as input and outputs a engine activation policy. The vehicle model and navigation map are learned from GPS traces with metadata, and includes the topological road structure, traversal speeds/times, battery consumption/regeneration, and ambient noise. We analyze our results in simulation on different navigation maps generated from actual GPS traces learned from a real series HEV. Experiments in simulation demonstrate that our approach compared with the baseline system can reduce total energy expenditure (EE), namely on hills, by up to 3%; total additional noise (AN) generated by up to 15%; and total mode transition (MT) frequency by up to 12%. The approach is demonstrated on a real series hybrid vehicle, driving on real public roads.**

I. INTRODUCTION

Hybrid electric vehicles (HEVs) benefit from the many energy and performance characteristics of pure electric vehicles as well as increased ranges and the rapid refueling of pure gas-powered vehicles [1]. There are three categories of HEV: series, parallel, and series-parallel [2]. Series HEVs have a straight-forward drivetrain that uses an efficient gas-powered engine whose only purpose is to power a compact battery that an electric motor uses to control the wheels. Conversely, both parallel and series-parallel have complex drivetrains that incorporate a gas-powered engine to directly control the wheels in addition to its electric motor. The key to reducing *energy expenditure (EE)*, *additional noise (AN)*, and *mode transitions (MT)* in HEVs, especially series HEVs, is designing an intelligent algorithm that activates and deactivates the gas-powered engine.

Engine activation approaches for series HEVs vary widely from hand-tuned control to machine learning techniques [1], [2], [3]. Early approaches use hand-crafted rules based on state-of-charge (SOC) of the battery under different conditions [4]. While simple, they are brittle and cannot benefit from predictions. One-step optimization methods that model the vehicle's battery, motor, and engine using deterministic predictions to make activation decisions have been tried [5]. Unfortunately, they are not robust to the uncertainties of the real world and can still be computationally challenging to solve in real time. A more recent hierarchical approach used high-level plans for optimal SOC trajectory values and low-level model-predictive control (MPC) based on predicted velocity and planned SOC [6]. Unfortunately, MPC is myopic and does not properly account for the effects of previously chosen actions on the present state.

Alliance Innovation Lab Silicon Valley, Santa Clara, CA 95051, USA
{kyle.wray,richard.lui,liam.pedersen}@nissan-usa.com

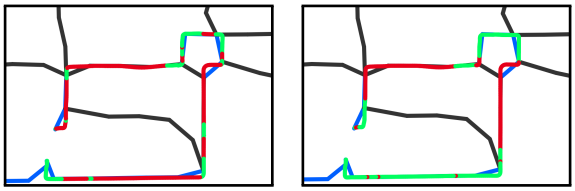

Fig. 1. Results from experiments on a real vehicle driving on public roads: the activation decisions of a 2018 Nissan Serena (left) versus the proposed approach (right). The green (engine off) and red (engine on) lines denote the vehicle's traversed path. This is overlaid on top of the navigation map in black (higher speed road) and blue (lower speed road).

Recently the focus has broadened to consider multiple objectives, including noise reduction [7] and emissions [8]. Noise reduction is increasingly cited as an important objective for customers and in society [9], although to date only a few hand-crafted or genetic algorithm approaches exist for it [7]. Another objective of rising import is minimizing mode transitions between engine activation and deactivation [10], as it is cited as potentially disruptive to the passengers if frequently applied [11]. Multi-objective methods to date have mainly focused on MPC approaches [12] with limited modeling of sequentiality in their optimization. The multi-objective Markov decision process (MDP) as a model have been used successfully in other vehicle-related domains such as route planning [13], [14] and autonomous vehicle decision-making [15], [16]. Prior work using Markov chains [8], [10] objectives show promise in Markov-based models, but do not propose a multi-objective MDP-based model built from learned navigation map and implemented on a real vehicle.

Specifically, this paper introduces a novel multi-objective stochastic shortest path (MOSSP) MDP-based model for intelligent engine activation. Based on the rising multi-objective foci in the literature discussed above, the objectives considered are to minimize the total energy expenditure, the experienced noise, and the number of mode transitions. The MOSSP is built on top of a learned navigation map from real vehicle data, customizable based on driver behavior.

Our contributions are: (1) a multi-objective SSP-based model for intelligent engine activation planning, (2) two novel multi-objective extensions of a state-of-the-art SSP solver called FLARES, both using a scalarization function and through local action restriction, (3) a description of a learning method for both vehicle parameters and the underlying navigation map, and (4) experiments demonstrating the effectiveness of the MOSSP engine activation approach in simulation and on an actual series hybrid vehicle (Figure 1).

II. BACKGROUND

A *multi-objective stochastic shortest path (MOSSP)* problem [17], [18] is defined by the tuple $\langle S, A, T, \vec{C}, s_0, G\rangle$. SSPs are a formal generalization of both finite and infinite horizon MDPs. S is a set of states. A is a set of actions. $T : S \times A \times S \rightarrow [0, 1]$ is a state transition function such that $T(s, a, s') = Pr(s' \mid s, a)$ denotes the probability that successor s' occurs given action a was performed in state s. $\vec{C} : S \times A \rightarrow \mathbb{R}^k$ is a multi-cost function such that $\vec{C}(s, a) = [C_1(s, a), \ldots, C_k(s, a)]^T$ with each $C_i(s, a)$ denoting the cost of objective i for being in a state s and performing the action a. The initial state s_0 is provided, as well as a set of goal states $G \subseteq S$.

A *policy* $\pi : S \rightarrow A$ maps each state to an action. Given a policy π, the *value* $\vec{V} : S \rightarrow \mathbb{R}^k$ defines the expected costs. For the initial state s_0, the value is:

$$\vec{V}(s_0) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \vec{C}(S_t, A_t)\Big| S_0{=}s_0; \pi\Big]$$

with S_t and A_t denoting the random variable of the state and action at time step t, respectively. A *scalarization function* $f : \mathbb{R}^k \rightarrow \mathbb{R}$ converts a MOSSP to an SSP [18] and is denoted with the subscript f such that $f(\vec{V}(s)) = V_f(s)$ and $f(\vec{C}(s, a)) = C_f(s, a)$. The *scalarized objective* is to find a policy π that minimize the expected costs of the initial state, governed by the Bellman optimality equation:

$$V_f(s) = \min_{a \in A} \Big(C_f(s, a) + \sum_{s' \in S} T(s, a, s')V_f(s')\Big). \quad (1)$$

In general, to solve a MOSSP requires a *proper policy* exists. A proper policy π has: (1) there exists a policy that reaches a goal with probability 1, and (2) all states that do not reach the goal with probability 1 result in an infinite cost.

A MOSSP can be solved by applying Equation 1 until convergence. However, this is very slow in general. More advanced optimal algorithms leverage heuristic search and Monte carlo sampling in order to intelligently explore the state space. One such optimal algorithm is called labeled real-time dynamic programming (LRTDP) [19]. A fast approximation of LRTDP is called fast labeling from residuals using sampling (FLARES) [20]. Its labeling technique checks a fixed depth instead of the full depth.

An alternative objective is to consider a lexicographic ordering over objectives [13] parameterized by slack $\vec{\delta} = [\delta_1, \ldots, \delta_{k-1}]^T$. Each objective i is maximized in order, allowing a deviation from its optimal value by up to δ_i to improve subsequent objectives. Formally, the *lexicographic objective* is to find a policy π such that for all i:

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & V_i^{\pi}(s_0) \\ \text{subject to} \quad & V_j^*(s_0) - V_j^{\pi}(s_0) \leq \delta_j, \qquad \forall j < i \end{aligned}$$

with each $V_j^*(s_0)$ denoting the optimal policy constrained in the same manner at j. A fast approximate of this is called local action restriction (LAR) [13]. It restricts the available actions via local slack $\vec{\eta}{=}[\eta_1, \ldots, \eta_{k-1}]^T$ at s by $A_{i+1}(s){=}\{a{\in}A_i(s) | V_i^*(s){-}Q_i^*(s, a){\leq}\eta_i\}$ with $A_1(s){=}A$.

III. INTELLIGENT ENGINE ACTIVATION PLANNING

We present the solution by describing vehicle parameters, the process for learning the navigation map, and then the MOSSP model built from the parameters and map.

A. Vehicle Parameters

The vehicle parameters are any relevant vehicle-specific information used by in the MOSSP model for planning. They are listed in Table III-A. Series hybrid vehicles have a small battery capacity θ_{bc} that charges from a gasoline-powered engine at power θ_{ep}. When activated, the engine generates noise following a learned model θ_{en} (Section III-C.3). The engine can be manually activated within the range $[\underline{\theta}_{ae}, \overline{\theta}_{ae}]$. However, if the battery level falls below this range then the engine turns on. Above this range turns the engine off. Also, the engine automatically turns on when traveling at speeds greater than θ_{ae}. In any case, when the engine is on, it expends fuel at an efficiency θ_{fe}, consuming at a rate of θ_{fc} from its fuel tank with capacity θ_{ft}.

Name	Units	Variable
Battery Capacity	kWh	θ_{bc}
Auto Engine On/Off Battery Level Range	[kWh, kWh]	$[\underline{\theta}_{ae}, \overline{\theta}_{ae}]$
Auto Engine On/Off Speed	km/h	θ_{ae}
Engine Battery Charge Power	kW	θ_{ep}
Engine Fuel Tank Capacity	L	θ_{ft}
Engine Fuel Consumption	L/100km	θ_{fc}
Engine Fuel Efficiency	%	θ_{fe}
Engine Noise	dB	θ_{en}

TABLE I
THE VEHICLE PARAMETERS RELEVANT TO PLANNING.

The vehicle parameters are assumed to be known, though they can be learned as well. The simplest form of learning is to learn a constant for each of these by averaging each term over time. A more expressive and accurate model fits a function to each (a generalization of a constant average). For example, engine battery charge power θ_{ep} is better described as function of the current battery level, speed, and slope.

B. Navigation Map

The navigation map is defined as the directed graph $\langle V, E\rangle$. Each vertex $v \in V$ simply has the parameters: id, latitude, longitude, and altitude, as defined in Table III-B. It defines a coordinate in space as well as a unique id. Each edge $e \in E$ has the parameters defined in Table III-B. It defines a to and from vertex, a unique id, and all the semantic road traversal information necessary for planning. Edge parameters are self-descriptive, such as the number of times traversed e_{ntt} and average speed e_{as}. Average battery consumption/regeneration e_{abcr} refers to all non-stop driving along the edge. It automatically incorporates the consequences of slope, road type, and even traffic, by simply recording on average how much change in battery level there was after traversal. It also independently models full stops, denoting how many times a stop occurred e_{nts}, the duration

Name	Units	Variable
Id	—	v_{id}
Latitude	degrees	v_{lat}
Longitude	degrees	v_{lon}
Altitude	m	v_{alt}

TABLE II

THE NAVIGATION MAP VERTEX PARAMETERS RELEVANT TO PLANNING.

Name	Units	Variable
id	—	e_{id}
From Vertex Id	—	e_{from}
To Vertex Id	—	e_{to}
Number of Times Traversed	—	e_{ntt}
Number of Times Stopped	—	e_{nts}
Average Speed	km/h	e_{as}
Average Traversal Time	h	e_{att}
Average Stop Time	h	e_{ast}
Average Battery Consumption/Regeneration	kWh	e_{abcr}
Average Battery Regeneration On Stop	kWh	e_{abrs}

TABLE III

THE NAVIGATION MAP EDGE PARAMETERS RELEVANT TO PLANNING.

of the stop e_{ast}, and how the average battery level change from regenerative braking e_{abrs}.

The navigation map can be learned from the vehicle traversing the roads. Let a *GPS trace* be defined as a vector $\vec{g} = \langle g_1, \ldots, g_{|\vec{g}|} \rangle$. Let G be a set of GPS traces. For each GPS trace, which are discrete points, we pair each $\vec{g}_i, \vec{g}_j \in G$ for each contiguous length that they are within a pre-defined tolerance $d_{tol} > 0$ (in meters) from one another. The average of the beginning and end points in this segment of $\vec{g}_i$ and $\vec{g}_j$ forms two vertices. An edge is then added that contains the parameters from Table III-B. That is, it averages all recorded speeds, battery consumption, etc. along these segments.

C. Intelligent Engine Activation MOSSP

The objective is to plan an engine activation for each possible state of the world that could arise. As the vehicle drives this plan can be further refined. This means given the vehicle's current position, plan if the gas-powered engine should be turned on or off given the historic driving patterns learned in the navigation map, including the final goal locations. Importantly, battery consumption/regeneration is stochastic based on: (1) the branching statistical distribution of edge traversal times; (2) multiple possible routes splitting and joining to reach the same goal, and (3) regenerative braking during stochastic stops in slow traffic and traffic lights. Thus the battery level at any upcoming navigation map edge has a probability distribution associated with it. This stochastic process is naturally modeled as a Markov chain; however, we also take actions that affect the battery level, namely to turn on or off the gas-powered engine. We propose a MOSSP, which is a generalization of a Markov decision process, that accurately models this planning process.

1) State and Action Spaces: The state space in the MOSSP is defined as $S = S_e \times S_{bl} \times S_{es}$. $S_e = E$ is the set of edges in the navigation map, that is the roads that the vehicle has historically driven. $S_{bl} \subset [0, \theta_{bc}]$ is the current battery level (in kWh) discretized at a regular interval. Here we consider a discretization resolution of 30 with $S_{bl} = \{\overline{\theta}_{bc}\frac{0}{30}, \overline{\theta}_{bc}\frac{1}{30}, \ldots, \overline{\theta}_{bc}\frac{30}{30}\}$. $S_{es} = \{\text{off}, \text{on}\}$ is the current engine status, either off or on.

The action space $A = A_{ea} = \{\text{off}, \text{on}\}$ is the engine activation choice: if the gas-powered engine should be turned off or on for the next edge traversed.

The initial state $s_0 = \langle s_{0e}, s_{0bl}, s_{0es} \rangle$ is the starting vehicle location edge $s_{0e} \in E$, battery level $s_{0bl} \in S_{bl} \subset [0, \theta_{bc}]$, and the engine status s_{0es} such as $s_{0es} = \text{off}$.

The goal states $s_g = \langle s_{ge}, s_{gbl}, s_{ges} \rangle \in G$ are any state that has a self-looping edge in the navigation map, denoting the end of learned GPS traces. Formally, the set of goal states are $G = \{s_g \in S | \exists e \in E \text{ s.t. } e_{to} = s_{ge} \wedge e_{from} = s_{ge}\}$.

2) State Transition Function: The transition function T requires capturing the movement in the navigation map's edges, the change in battery level, and the change in engine status based on the map and action performed. For any normal state $s \in S - G$ that is not a goal, the state transition for an action a has three components:

$$T(s, a, s') = T_e(s, a, s') T_{bl}(s, a, s') T_{es}(s, a, s')$$

The first component governs how the location changes stochastically following the learned navigation map's topological graph formed by various from e_{from} to e_{to}. To this end, let the neighboring successor edges (roads) of state s be $N(s) = \{e \in E | e_{from} = s_{e,to}\}$. Formally, T_e is then:

$$T_e(s, a, s') = \begin{cases} \frac{s'_{e,ntt}}{\sum_{e \in N(s)} e_{ntt}}, & \text{if } s_{e,to} = s'_{e,from} \\ 0, & \text{otherwise} \end{cases}$$

Intuitively, the planner considers the likelihood of navigating on a successor road given the current road to be equal to the learned patterns of routes that the driver has historically driven in the past.

The second component governs how the battery level changes stochastically following the learned navigation map edge s_e's energy consumption/regeneration $s_{e,abcr}$, average traversal time $s_{e,att}$, engine activation action $a \in A$, the engine's battery charge power θ_{ep}, the learned number of times stopped $s_{e,nts}$, the learned number of traversal times $s_{e,ntt}$, and the average regeneration on a stop $s_{e,abrs}$. Due to the discretization of battery levels S_{bl}, we first must compute the exact modeled change in s^*_{bl}:

$$s^*_{bl} = \begin{cases} s_{bl} + s_{e,abcr} + \frac{s_{e,nts}}{s_{e,ntt}} s_{e,abrs}, & \text{if } s_{es} = \text{off} \\ s_{bl} + s_{e,abcr} + \frac{s_{e,nts}}{s_{e,ntt}} s_{e,abrs} + \frac{s_{e,att}}{s_{e,ntt}} \theta_{ep}, & \text{if } s_{es} = \text{on} \end{cases}$$

Using s^*_{bl} we probabilistically map this to a discrete s'_{bl} via:

$$T_{bl}(s, a, s') = \begin{cases} 1 - \frac{s^*_{bl} - \underline{s}^*_{bl}}{\overline{s}^*_{bl} - \underline{s}^*_{bl}}, & \text{if } s'_{bl} = \underline{s}^*_{bl} \\ \frac{s^*_{bl} - \underline{s}^*_{bl}}{\overline{s}^*_{bl} - \underline{s}^*_{bl}}, & \text{if } s'_{bl} = \overline{s}^*_{bl} \end{cases}$$

with $\underline{s}^*_{bl}, \overline{s}^*_{bl} \in S_{bl}$ being the nearest lower and upper battery level state factors from s^*_{bl} in S_{bl}, respectively.

The third component governs the change in engine status based on the action a, as well as the hardwired auto engine

on/off range $[\underline{\theta}_{ae}, \overline{\theta}_{ae}]$ and auto engine on/off speed θ_{ae}.

$$T_{es}(s,a,s')=\begin{cases} 1, & \text{if } (s_{e,as}{>}\theta_{ae} \wedge s'_{es}{=}\text{on}) \\ & \quad \vee (s_{e,as}{\leq}\theta_{ae} \wedge s_{bl}{<}\underline{\theta}_{ae} \wedge s'_{es}{=}\text{on}) \\ & \quad \vee (s_{e,as}{\leq}\theta_{ae} \wedge s_{bl}{>}\overline{\theta}_{ae} \wedge s'_{es}{=}\text{off}) \\ & \quad \vee (s_{e,as}{\leq}\theta_{ae} \wedge s_{bl}{\in}[\underline{\theta}_{ae}, \overline{\theta}_{ae}] \wedge s'_{es}{=}a) \\ 0, & \text{otherwise} \end{cases}$$

Intuitively, this just models the hardcoded engine behavior. The engine automatically turns on when the speed is high, a necessity when the electric motor draws a large amount of power. The engine automatically turns off or on when the battery capacity is too high or low, respectively, for long-term vehicle and battery health. Otherwise the engine follows activation requests via the MOSSP policy's action.

Lastly, for a goal state $s_g \in G \subset S$ and action a:

$$T(s_g, a, s') = \begin{cases} 1, & \text{if } s' = s_g \\ 0, & \text{otherwise} \end{cases}$$

which simply defines goals as a self-looping absorbing states.

3) Cost Functions: Multiple costs are considered in the MOSSP planning model. An important cost is energy expenditure (in kWh) which must factor in the battery consumption or regeneration of traversing an edge (road) in the navigation map, the expected energy gains of regenerative braking, and the cost of fuel spent to regenerate the battery when using the gas-powered engine. The toggling of the engine from off to on or on to off is also a cost, as this is both inefficient and can be jarring to the driver and passengers. A final cost to consider is how much extra noise (in dB) is generated beyond the learned ambient environment's average noise level. Ideally, all of these costs are factored into the objective function and minimized by the planner.

Let the cost vector $\vec{C}$ be defined for these objectives: $\vec{C}(s,a) = [C_{ee}(s,a), C_{an}(s,a), C_{mt}(s,a)]^T$.

The first component $C_{ee}(s,a)$ is the expected energy expenditure. To begin, we need to compute the contribution of the engine status (s_{es}, θ_{ep}, and $s_{e,att}$) and expected regeneration from any regenerative braking ($s_{e,ntt}$, $s_{e,nts}$, and $s_{e,abrs}$ from Table III-B), denoted as $\epsilon_{es}(s,a)$ and $\epsilon_{rb}(s,a)$, respectively. Formally:

$$\epsilon_{es}(s,a) = \begin{cases} \theta_{ep} s_{e,att}, & \text{if } s_{es} = \text{on} \\ 0, & \text{otherwise} \end{cases}$$

describes the expected power times expected time if the engine is on, and

$$\epsilon_{rb}(s,a) = \frac{s_{e,nts}}{s_{e,ntt}} s_{e,abrs}$$

describes the expected regeneration while stopping if the vehicles stops. Now we can compute the expected new battery level $b_l(s,a)$ and battery consumption $b_c(s,a)$:

$$b_l(s,a) = s_{bl} + s_{e,abcr} + \epsilon_{es}(s,a) + \epsilon_{rb}(s,a)$$
$$b_c(s,a) = \max\{0, -s_{e,abcr}\}.$$

The wasted energy $\epsilon_{we}(s,a)$ is then computed by any excess battery level that would have gone above the capacity

$$\epsilon_{we}(s,a) = \max\{0, b_l(s,a) - \theta_{bc}\}.$$

Given the battery level and wasted energy, we can compute the battery regeneration $b_r(s,a)$ amount:

$$b_r(s,a) = \max\Big\{0, \max\{0, s_{e,abcr}\} + \epsilon_{es}(s,a) + \epsilon_{rb}(s,a) - \epsilon_{we}(s,a)\Big\}$$

The total energy also includes how much fuel was spent regenerating the battery. This amount of energy contribution from fuel used to power the engine, $\epsilon_{fe}(s,a)$, is proportional to the ratio of engine's generated energy to the engine's efficiency θ_{fe} in conversion of fuel to energy stored in the battery:

$$\epsilon_{fe}(s,a) = \frac{\theta_{ep} s_{e,att}}{\theta_{fe}}.$$

Finally, we can compute the total energy cost $C_{ee}(s,a)$:

$$C_{ee}(s,a) = b_c(s,a) - b_r(s,a) + \epsilon_{fe}(s,a)$$

which is equal to the battery energy consumed minus the battery energy regenerated plus the fuel energy expended as part of any engine activation.

The important second component $C_{an}(s,a)$ is how much additional noise is generated if the engine is activated. A vehicle noise model $\theta_{en} : S_e \times S_{es} \to \mathbb{R}^+$ maps the speed limit on an edge and the engine status to the cabin noise (in dB). The model was learned using polynomial regression by driving the vehicle at different speeds (e.g., 10 km/h increments from 0 to 100) and recording the cabin noise (in dB) with the engine on and off. The amount of additional noise experienced (in dBh) is defined as the excess noise of the current engine status versus it being off, for the duration of travel time on the current road:

$$C_{an}(s,a) = (\theta_{en}(s_e, s_{es}) - \theta_{en}(s_e, \text{off}))s_{e,att}.$$

The third component $C_{mt}(s,a)$ is the likelihood of a mode transition (i.e., expected value of the engine switching). This is simply equal to the probability that the engine will toggle:

$$C_{mt}(s,a) = \sum_{s' \in S} T(s,a,s')[s_{es} \neq s'_{es}]$$

with Iverson bracket $[\cdot]$. This weighs the probability of the selected action a intentionally toggling the engine, as well as the automatic toggling of the engine as governed by θ_{ae} and $[\underline{\theta}_{ae}, \overline{\theta}_{ae}]$.

Lastly, for any goal state $s_g \in G$, the cost for all actions a is zero $\vec{C}(s_g, a) = 0$, completing the requirement of a goal.

IV. EXPERIMENTS

The intelligent engine activation model and planner is evaluated using real navigation maps generated from an actual 2018 Nissan Serena, comparing baseline, scalarization, and lexicographic algorithms. This section covers the experimental setting, the results, and a discussion.

Navigation Map	Route	Rule-Based Baseline			Scalarization (f_1)			Scalarization (f_2)			Lexicographic ($\vec{C}_1$, $\vec{\eta}_1$)			Lexicographic ($\vec{C}_2$, $\vec{\eta}_2$)		
		V_{ee}	V_{an}	V_{mt}	V_{ee}	V_{an}	V_{mt}	V_{ee}	V_{an}	V_{mt}	V_{ee}	V_{an}	V_{mt}	V_{ee}	V_{an}	V_{mt}
Daily Driving	1	2.87	0.19	1.90	1.46	0.04	2.63	1.49	0.04	2.63	1.55	0.05	1.33	1.89	0.06	0.95
	2	3.32	0.15	1.99	1.84	0.06	3.87	1.87	0.06	3.95	1.92	0.06	2.32	2.04	0.07	1.73
	3	2.33	0.12	1.62	0.80	0.03	1.28	0.81	0.03	1.35	0.85	0.03	0.91	1.12	0.05	1.31
	4	5.16	0.20	2.99	4.21	0.13	10.25	4.24	0.13	10.05	4.33	0.14	3.56	4.49	0.15	2.65
Mountain Driving	5	18.36	0.62	7.87	17.48	0.65	36.89	17.52	0.66	36.78	17.56	0.60	9.61	17.58	0.57	7.47
	6	23.79	0.80	9.91	22.28	0.74	46.19	22.32	0.75	46.03	22.61	0.79	13.34	22.63	0.72	10.18
	7	18.36	0.62	7.87	17.48	0.65	36.89	17.52	0.66	36.78	17.56	0.60	9.75	17.57	0.57	7.49
	8	32.75	0.89	6.73	31.88	0.85	49.19	31.88	0.85	48.93	31.97	0.79	13.20	32.01	0.83	11.67

TABLE IV

RESULTS FROM EXPERIMENTS FOR: (1) THE RULE-BASED BASELINE, AND THE PROPOSED MOSSP WITH FOUR ALGORITHMS: (2 & 3) SCALARIZATION AND (4 & 5) LEXICOGRAPHIC. TWO NAVIGIATION MAPS ARE CONSIDERED: (1) DAILY DRIVING AND (2) MOUNTAIN DRIVING. FOUR DISINCT ROUTES ARE TRAVERSED WITHIN EACH MAP. METRICS ARE THE THREE OBJECTIVES AT THE INITIAL STATE: (1) ENERGY EXPENDITURE V_{ee}, (2) ADDITIONAL NOISE V_{an}, AND (3) MODE TRANSITIONS V_{mt}. EACH REPRESENT THE AVERAGE FROM 1000 SIMULATIONS.

A. Vehicle Parameters

The vehicle parameters assign the battery capacity to θ_{bc} to the Serena's capacity and the automatic engine on/off range to $[\underline{\theta}_{ae}, \overline{\theta}_{ae}] = [0.4\theta_{bc}, 0.8\theta_{bc}]$. This automatic engine on/off range was determined by repeatedly pressing the *charge mode* and *manner mode* buttons until engine behavior changed at a specific SOC.

B. Navigation Maps

Real vehicle data was collected from a 2018 Nissan Serena in the Silicon Valley area in California, USA to learn two distinct navigation maps. The GPS and CAN data were recorded, building a model of 100-500 meter road increment along the traces. The traces overlap with one another and were combined to form each of the maps. These two navigation maps illustrate our MOSSP approach on real world series hybrid vehicle data. They incorporate a wide array of features, ranging from different traversal speeds (e.g., including traffic effects) to varied battery consumption/regeneration amounts (e.g., increased power consumption on hills and downhill regenerative braking).

The *daily driving navigation map* is built from both residential and business roads. It represents multiple routes driven to go between work and home, as well as nearby businesses such as a grocery store. The roads consist of both low suburban speeds of 30 km/h and higher highway speeds of 80 km/h, as well as traffic and stops at traffic lights.

The *mountain driving navigation map* is built primarily from more rural mountainous and winding hilly roads. It represents a rural trip and the effects of mixed rural speeds, turns, and steep elevation changes on a vehicle that ascends and descends. The roads consist of hilly turns and speeds of around 50 km/h with an altitude change of up to 300 meters.

The navigation maps are visualized in Figure 1 and the left columns of Figures 2 and 4. As detailed above, they are built from sampled traces along the roads. Slower speed "quieter roads" are emphasized in blue. These roads have more engine noise would be experienced here versus other higher speed roads.

C. Algorithms

Both *scalarization FLARES* and *lexicographic FLARES* are implemented for the novel MOSSP engine activation model. Two scalarizations are used f_1 and f_2:

$$(1)\ f_1(\vec{C}(s,a)) = C_{ee}(s,a) + 0.1C_{an}(s,a) + 0.1C_{mt}(s,a)$$
$$(2)\ f_2(\vec{C}(s,a)) = 0.1C_{ee}(s,a) + C_{an}(s,a) + 0.1C_{mt}(s,a)$$

Two lexicographic orderings with slacks are used:

$$(1)\ \vec{C}_1 = [C_{ee}, C_{an}, C_{mt}]^T \text{ with } \vec{\eta}_1 = [3, 0.2]^T$$
$$(2)\ \vec{C}_2 = [C_{an}, C_{ee}, C_{mt}]^T \text{ with } \vec{\eta}_2 = [0.1, 0.2]^T$$

The proposed MOSSP-based approach is evaluated on these four solvers and compared to a *rule-based baseline* [4] and the actual *Serena (2018) baseline*. Its parameters are based on the analyzed activation behavior of a 2018 Nissan Serena. The Serena has two manual buttons to control the engine (see Figure 3). This baseline and the vehicle parameters were determined by driving the Serena at various speeds while pressing these manual buttons. It follows the behavior set by the automatic engine on/off range $[\underline{\theta}_{ae}, \overline{\theta}_{ae}]$ and automatic engine on speed θ_{ae}.

D. Experiments

Experiments were done both in simulation and on a real vehicle driving on public roads.

The simulation experiments were run on an Intel Core i9-9980HK CPU at 2.40GHz with 8 cores, 32GB of RAM, and Ubuntu 16.04 with Linux kernel 4.15. The simulation environment and algorithms were written in Julia 1.5.2. For each algorithm, navigation map, and route, 1000 simulations were executed. Table IV summarizes these results. Figures 2 and 4 shows the behaviors of each algorithm over time.

The real world experiments were run on the 2018 Nissan Serena in California. The algorithms control the Serena electronically through an interface to the manual engine control buttons on the vehicle's dashboard. Figure 3 shows the experimental test vehicle used. Figure 1 shows the results of experiments on real Serena driving on real public roads.

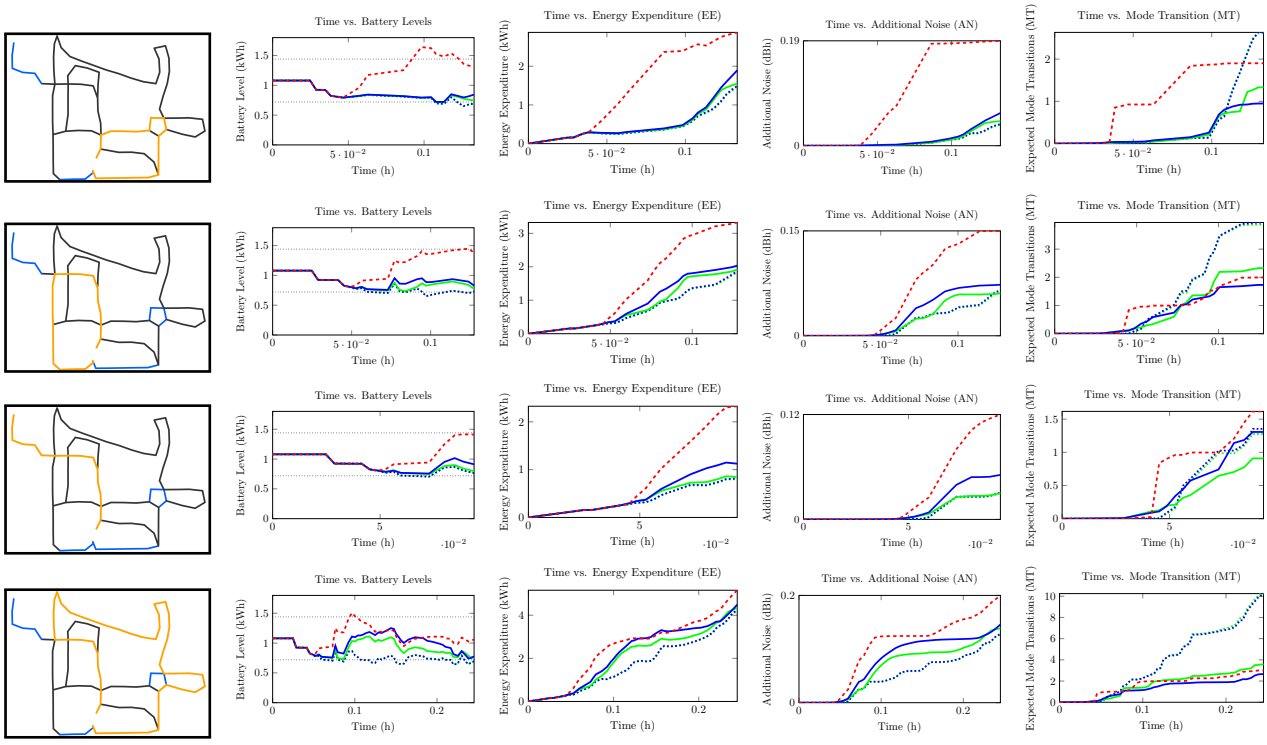

Fig. 2. Results from *daily driving navigation map* simulations. Each row is a route: 1, 2, 3, and 4. The first column shows the route overlaid in orange on the navigation map. The next columns show the average battery level, energy expenditure, additional noise, and mode transitions, over time. The five algorithms average performance is shown in lines: rule-based baseline (red), scalarization (f_1) & (f_2) (solid blue & green), lexicographic $(\vec{C}_1, \vec{\eta}_1)$ & $(\vec{C}_2, \vec{\eta}_2)$ (dotted blue & green). Each individual line is the average over 1000 simulations. The navigation map is derived from real vehicle data.

E. Discussion

Table IV shows the benefits of the proposed MOSSP approach over the baseline, and highlights the difference between the scalarization versus lexicographic modeling. The MOSSP has up to 2 kWh less expected total energy expenditure than the baseline. Accounting for variations in the final battery levels (Figures 2 and 4), the realized savings is up to 1 kWh. Importantly, the MOSSP significantly reduces the experienced noise, halving it in many cases.

Figure 1 shows the success of the proposed approach over the actual 2018 Nissan Serena baseline. The MOSSP approach proactively plans to regenerate battery on higher speed roads where the experienced engine noise can better be masked, and disables the engine on lower speed roads, all while maintaining solid energy efficiency. Conversely, the vehicle baseline is much more reactive, myopically enabling the engine based solely on SOC.

Comparing Figure 2 versus Figure 4 highlights the effect of the type of driving on engine activation algorithms. Mountain routes are longer and require more energy, resulting in modest but predictable energy savings. The MOSSP algorithms are able to predict uphill and downhill sections, proactively regenerating and conserving the battery.

Both scalarization and lexicographic objectives are shown to work; the former slightly outperforms the latter, at the cost many more mode transitions. In both cases, the noise and energy preference induces the corresponding desired engine behavior. This illustrates the MOSSP's flexibility, allowing direct customization for different preferences.

Fig. 3. The 2018 Nissan Serena vehicle used in the experiments (top). The manual control buttons are also highlighted in green (bottom left). They are triggered electronically in the experiments, such as activating charge mode (bottom right). Figure 1 shows experimental results on this real vehicle.

V. CONCLUSION

This paper demonstrates the success of using stochastic shortest path (SSP) problems, a generalized MDP-based model, as a foundation of intelligent engine activation planners for series hybrid vehicles. Experiments show that the proposed MOSSP engine activation model, created from real vehicle data, using the FLARES algorithm outperforms rule-based approaches on a real series hybrid vehicle.

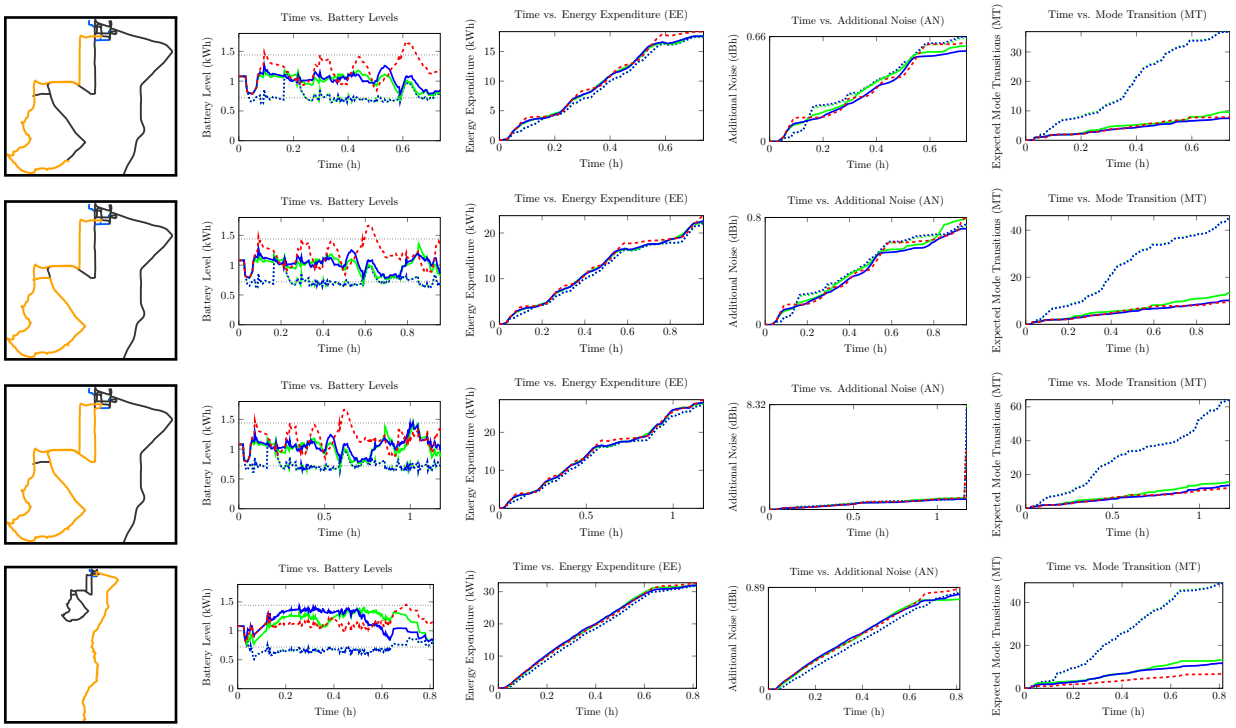

Fig. 4. Results from *mountain driving navigation map* simulations. Each row is a route: 5, 6, 7, and 8. The first column shows the route overlaid in orange on the navigation map. The next columns show the average battery level, energy expenditure, additional noise, and mode transitions, over time. The five algorithms average performance is shown in lines: rule-based baseline (red), scalarization (f_1) & (f_2) (solid blue & green), lexicographic $(\vec{C}_1, \vec{\eta}_1)$ & $(\vec{C}_2, \vec{\eta}_2)$ (dotted blue & green). Each individual line is the average over 1000 simulations. The navigation map is derived from real vehicle data.

REFERENCES

[1] C. M. Martinez, X. Hu, D. Cao, E. Velenis, B. Gao, and M. Wellers, "Energy management in plug-in hybrid electric vehicles: Recent progress and a connected vehicles perspective," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4534–4549, 2016.

[2] E. Silvas, T. Hofman, N. Murgovski, L. P. Etman, and M. Steinbuch, "Review of optimization strategies for system-level design in hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 57–70, 2016.

[3] P. Zhang, F. Yan, and C. Du, "A comprehensive analysis of energy management strategies for hybrid electric vehicles based on bibliometrics," *Renewable and Sustainable Energy Reviews*, vol. 48, pp. 88–104, 2015.

[4] S. Overington and S. Rajakaruna, "High-efficiency control of internal combustion engines in blended charge depletion/charge sustenance strategies for plug-in hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 48–61, 2014.

[5] P. Elbert, T. Nüesch, A. Ritter, N. Murgovski, and L. Guzzella, "Engine on/off control for the energy management of a serial hybrid electric bus via convex optimization," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 8, pp. 3549–3559, 2014.

[6] Z. Chen, N. Guo, J. Shen, R. Xiao, and P. Dong, "A hierarchical energy management strategy for power-split plug-in hybrid electric vehicles considering velocity prediction," *IEEE Access*, vol. 6, pp. 33 261–33 274, 2018.

[7] M. Aliramezani, M. Khadem Nahvi, and M. Delkhosh, "Optimal energy management strategy of a hybrid electric vehicle considering engine noise," *Journal of Vibration and Control*, vol. 24, no. 23, pp. 5546–5555, 2018.

[8] A. A. Malikopoulos, "A multiobjective optimization framework for online stochastic optimal control in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 2, pp. 440–450, 2016.

[9] Y. Qin, X. Tang, T. Jia, Z. Duan, J. Zhang, Y. Li, and L. Zheng, "Noise and vibration suppression in hybrid electric vehicles: State of the art and challenges," *Renewable and Sustainable Energy Reviews*, vol. 124, p. 109782, 2020.

[10] L. Li, S. You, and C. Yang, "Multi-objective stochastic MPC-based system control architecture for plug-in hybrid electric buses," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 8, pp. 4752–4763, 2016.

[11] H. Kim, J. Kim, and H. Lee, "Mode transition control using disturbance compensation for a parallel hybrid electric vehicle," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 225, no. 2, pp. 150–166, 2011.

[12] Y. Zhou, A. Ravey, and M.-C. Péra, "Multi-objective energy management for fuel cell electric vehicles using online-learning enhanced markov speed predictor," *Energy Conversion and Management*, vol. 213, p. 112821, 2020.

[13] K. H. Wray, S. Zilberstein, and A.-I. Mouaddib, "Multi-objective MDPs with conditional lexicographic reward preferences," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015, pp. 3418–3424.

[14] K. H. Wray and S. Zilberstein, "Multi-objective POMDPs with lexicographic reward preferences," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, 2015, pp. 1719–1725.

[15] K. H. Wray, S. J. Witwicki, and S. Zilberstein, "Online decision-making for scalable autonomous systems," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 4768–4774.

[16] K. H. Wray, "Abstractions in reasoning for long-term autonomy," Ph.D. dissertation, University of Massachusetts, Amherst, MA, 2019.

[17] D. P. Bertsekas and J. N. Tsitsiklis, "An analysis of stochastic shortest path problems," *Mathematics of Operations Research*, vol. 16, no. 3, pp. 580–595, 1991.

[18] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[19] B. Bonet and H. Geffner, "Labeled RTDP: Improving the convergence of real-time dynamic programming," in *Proceedings of the 13th International Conference on Automated Planning and Scheduling*, 2003, pp. 12–21.

[20] L. Pineda, K. H. Wray, and S. Zilberstein, "Fast SSP solvers using short-sighted labeling," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2017, pp. 3629–3635.