Chapter 8
Eco-Driving Practical Implementation

The aim of this section is to present a few practical implementation issues of eco-driving. In Sect. 8.1 we shall discuss the various eco-driving systems that implement partly or fully the concepts described in the previous chapters. All of these systems use the localization, perception, and planning/control functions that have been treated throughout this book. A few additional and specific algorithmic and implementation issues are further detailed in Sect. 8.2. Then, we will discuss in Sect. 8.3 the issues related to advising a human driver about the optimal speed to follow and the direct implementation of eco-driving via an autonomous driving system.

8.1 Implementation of Eco-Driving Concepts

In this section we present, in order of increasing complexity and comprehensiveness, eco-coaching (Sect. 8.1.1), PCC (8.1.2), eco-ACC (Sect. 8.1.3), and the most general predictive eco-driving schemes (Sect. 8.1.4).

8.1.1 Eco-Coaching

Eco-coaching is the online assessment of the driving style of human drivers by comparing their actual speed traces to the energy-optimal speed profiles. An example of eco-coaching system that has been demonstrated for EVs and ICEs is Geco [1, 2]. Its practical implementation follows a number of steps as illustrated by the flowchart in Fig. 8.1.

A. Sciarretta and A. Vahidi, *Energy-Efficient Driving of Road Vehicles*,
Lecture Notes in Intelligent Transportation and Infrastructure,
https://doi.org/10.1007/978-3-030-24127-8_8

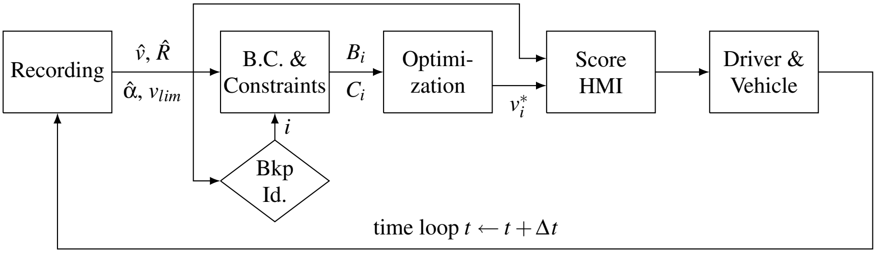


Fig. 8.1 Conceptual sketch of an eco-coaching system

The system is based on the real-time recording of the actual vehicle speed $\hat{v}$. This measure can be obtained from several sources, including integration of GPS coordinates and on-board measurements (odometry) available on the vehicle communication bus. Relying only on GPS-based measurement makes the system free from in-vehicle connections and able to run on a mobile device or as a web service, but suffers from inaccuracies due to satellite signal quality. However, the current precision of GNSS is considered sufficient for this type of application, and accuracy is expected to further improve with the anticipated trend in localization precision. In addition to speed, also curvature $\hat{R}$, slope $\hat{\alpha}$, and legal speed limit v_{lim} need to be extracted from GPS and map data, e.g., using algorithms introduced in Sect. 8.2.1.

These speed traces are continuously analyzed to identify when a *breakpoint* occurs. As explained in Sect. 8.2.2, such identification is based on the definition of breakpoints either as the locations where the characteristics of the road change, or the points where surrounding traffic induces strong decelerations or stops.

When a breakpoint is detected, a sub-trip is defined encompassing the last two breakpoints, see Fig. 8.2. For this sub-trip, the distance covered, the travel time, and the average speed are evaluated as

$$t_f^{(i)} = \tau_{i+1} - \tau_i; \quad s_f^{(i)} = \int_{\tau_i}^{\tau_{i+1}} \hat{v}(\tau) d\tau; \quad v_i^{(i)} = \hat{v}(\tau_i); \quad v_f^{(i)} = \hat{v}(\tau_{i+1}) \,, \qquad (8.1)$$

where τ is time of record, τ_{i+1} is current time and τ_i is the time of last breakpoint. These quantities form the boundary condition vector $B_i \triangleq \{t_f^{(i)}, s_f^{(i)}, v_i^{(i)}, v_f^{(i)}\}$, while the road parameter vector $C_i \triangleq \{\alpha^{(i)}, R^{(i)}, v_{lim}^{(i)}\}$ is composed of the slope, curvature, and speed limit in the sub-trip.

A speed optimization is then performed using the elements of B_i and C_i as the optimization parameters. The methods described in Chap. 6 can be generally used to optimize the speed profile and compute the minimal energy E_T^*. For EVs, the simpler approach is to use the parabolic speed profile (6.74). Alternatively, optimal speed profiles can be obtained by training neural networks to evaluate the optimal sequence of control modes and their switching times, that is, provide the vector $O_i = \{t_1, \ldots\}$ as a function of the input vector I_i as also described in Chap. 6.

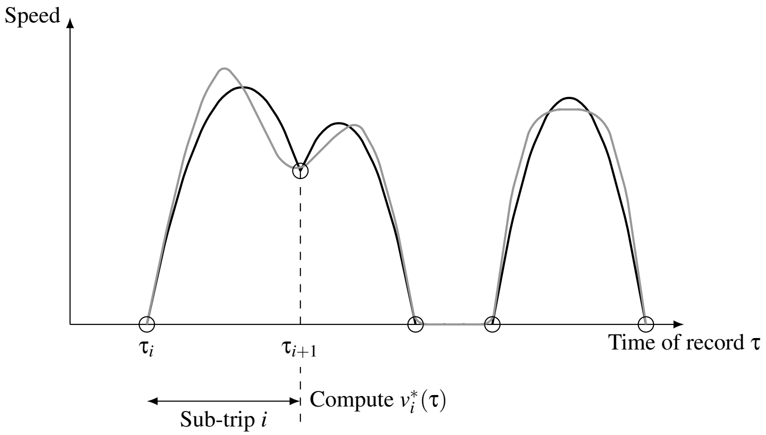


Fig. 8.2 Illustration of the eco-coaching approach, with the recorded speed (black), the breakpoints (circles), a sub-trip, the time instant at which an optimization is performed, and the optimized speed profile (gray)

The output of such a process is ultimately the optimal speed profile $v_i^*(\tau)$, $\tau \in [\tau_i, \tau_{i+1}]$ that the driver should have followed during the last sub-trip, with the same constraints in terms of duration, distance, initial and final speed, maximal speed, than the actual profile performed.

The optimal speed profile is typically displayed to the drivers by an HMI (see Sect. 8.3.1), in order to make them aware of the best driving practices. Additionally, the system can provide an “eco-driving” score, comparing a model-based estimation of the actual (tank) energy consumed during the sub-trip, with the optimum calculated. The drivers are expected to learn from these scores and adapt their driving style toward the optimum.

8.1.2 Predictive Cruise Control

In this book we adopt the terminology of Predictive Cruise Control (PCC) to denote those cruise control systems that, in addition to the standard feature of tracking a reference speed, are mainly focused on anticipation of road slopes. A few such systems are actually on the market as of this book’s writing,[1] particularly for heavy-duty truck applications. As shown in Sect. 7.3, a speed profile that anticipates and reacts to road slopes in a certain way (reach a minimum speed at the top of a hill, maximum speed at the bottom of a downhill) may actually coincide with the minimal-energy speed profile.

[1] Introduced by Daimler Trucks in 2009, now PCC systems are available on trucks by DAF, Kenworth, Volvo Trucks, as well some Mercedes and Volkswagen cars, among others.

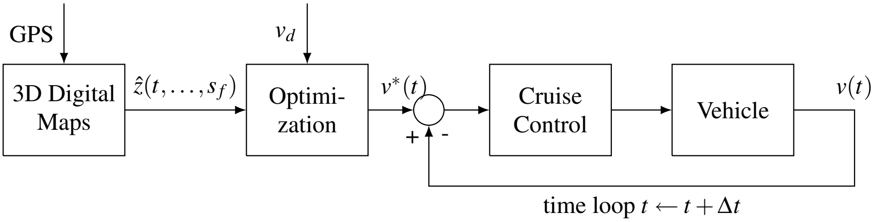


Fig. 8.3 Conceptual sketch of a predictive cruise control system

In a typical embodiment of PCC, see Fig. 8.3, the GPS signal is an input. Using stored or retrieved 3D digital road maps, PCC estimates the altitude profile of the road ahead, $\hat{z}(\sigma)$, $\sigma \in [s, s_f]$, up to the selected horizon s_f. In order to limit the computing resources, this horizon might be shorter than the whole trip. An algorithm then calculates the optimal speed profile[2] $v^*(\tau)$, $\tau \in [t, t_f]$, for the temporal horizon t_f that is linked to s_f through the desired average speed v_d. The first value of the optimal speed reference is sent to a speed controller (which is in fact a conventional cruise controller), which ultimately pilots the vehicle's actuators to track the reference speed as a function of the measured actual speed.

8.1.3 Eco-ACC

As briefly explained in Sect. 4.2.2, adaptive cruise controllers (ACC) are cruise control systems that automatically adjust the vehicle speed to keep a safe gap from the vehicle ahead. Besides the simple PI control schemes described in Sects. 3.3.4.1 and 4.2.2, ACC could in principle be implemented by solving, at each time, an optimal control problem that minimizes some safety-related cost. Those systems that are known as eco-ACC use in addition an energy-efficiency-related cost, whence comes their name.

Figure 8.4 shows a flowchart illustrating the concept. Unlike in PCC where a single optimization is performed at the beginning of an entire trip or of a long section, in an eco-ACC system a new optimization is performed at every time step for a prediction or look-ahead horizon. This approach is often generically referred to as Model Predictive Control (MPC), and is described more extensively in Sect. 8.2.5.

Typically, in eco-ACC systems the prediction horizon t_f is fixed, in an embodiment that is called a *receding horizon* approach. Over this horizon, the disturbances and constraints acting on the vehicle must be predicted. Considering for simplicity only road grade and curvature, top speed limit and preceding vehicle, the road parameter vector is $C(t) \triangleq \{\alpha(s(\tau)), R(s(\tau)), v_{lim}(s(\tau)), s_p(\tau)\}$, $\tau \in [t, t+t_f]$.

[2] For ICEVs, optimal gear profile is usually also calculated.

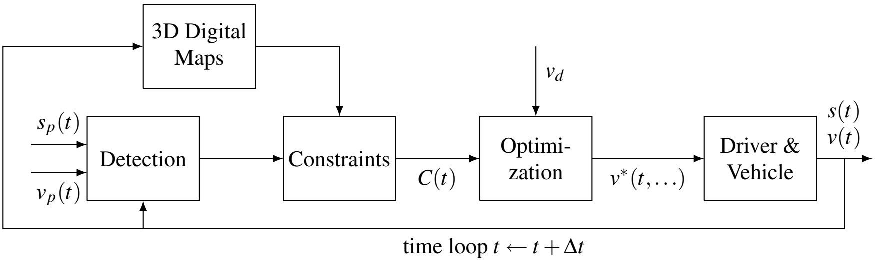


Fig. 8.4 Conceptual sketch of an Eco-ACC system

The road characteristics within the prediction horizon are retrieved from detailed 3D maps that are either stored within the system or accessed remotely. The presence of a preceding vehicle (moving or stationary obstacle), together with its relative position and speed to the host vehicle, is assumed to be detected thanks to ADAS sensors such those introduced in Chap. 3. Based on the current relative speed and headway, a dedicated algorithm estimates the leader position s_p, for the upcoming horizon. Such estimation is very critical since it strongly affects the effectiveness of the optimization results, and is described in some detail in Sect. 8.2.3. In addition, V2I communication or camera sensors can provide information on the state of upcoming traffic lights, which also affects the optimization process by introducing additional constraints. A few simple algorithms for traffic light detection are presented in Sect. 8.2.4.

Once the constraints are set, optimization of the speed profile of the host vehicle can be run for the entire prediction horizon. The cost function considers several factors. It generally includes the energy-based cost related to the tank energy, the safety cost related to the headway distance with the preceding vehicle, but might also include a regulation cost that penalizes the variations of the control inputs, and a speed smoothness cost that penalizes deviations from the desired average speed. The OCP is completed by the state equations and by the control and speed constraints, and can be summarized as

$$\begin{aligned}
\underset{\mathbf{u}(t)}{\text{minimize}} \quad & J = \int_t^{t+t_f} \Big[w_1 P_T(v(\tau), \mathbf{u}(\tau), \alpha(s(\tau))) + w_2 (s(\tau) - s_p(\tau))^2 + \\
& + w_3 (v(\tau) - v_d)^2 + w_4 (\mathbf{u}(\tau) - \mathbf{u}_d)^2 \Big] \, d\tau \\
\text{subject to} \quad & \dot{s}(\tau) = v(\tau) \\
& \dot{v}(\tau) = f(v(\tau), \mathbf{u}(\tau), \alpha(s(\tau))) \\
& \mathbf{u}_{min}(v(\tau), \tau) \leqslant \mathbf{u} \leqslant \mathbf{u}_{max}(v(\tau), \tau) \\
& 0 \leq v(\tau) \leq \min(v_{lim}(s(\tau)), v_{turn}(R(s(\tau))))
\end{aligned} \tag{8.2}$$

where $\mathbf{u}$ is the control vector, $\mathbf{u}_d$ is the steady-state control corresponding to cruising at constant speed v_d, f is the speed dynamics equation (6.15), and the w's are weighting factors chosen opportunely. The output of this optimization block is the optimal speed profile for the entire horizon $v^*(\tau)$, $\tau \in [t, t + t_f]$.

Finally, this speed profile is either advised to the human driver through an HMI (see Sect. 8.3.1) or directly realized through a vehicle controller in the case of an automated speed controller. The actual position and speed of the host vehicle serve as the inputs to update the boundary conditions. A new iteration is performed at each time step.

8.1.4 Predictive Eco-Driving

The eco-ACC concept presented in the last section is not the most general embodiment of eco-driving techniques. Particularly in urban driving, the receding fixed horizon approach must be replaced by a "shrinking" horizon based on a predictive breakpoint detection. In addition, a reference speed to track might not necessarily be available, and in the most general case the speed must be fully optimized.

The fundamentals of such predictive eco-driving have been presented and discussed in the previous chapters. Figure 8.5 shows a flowchart illustrating the practical implementation of an eco-driving function. Similarly to an eco-ACC system, the approach is intrinsically iterative, with a new optimization that is performed every time step for the whole extent of the prediction horizon.

Typically, this horizon is set as the remaining trip to the *next infrastructure* breakpoint (e.g., a signalized intersection) and must be defined by the distance to be covered $s_f(t)$ and an estimation of the travel time $t_f(t)$. These pieces of information can be obtained, e.g., from an eco-routing system or directly from real-time traffic and infrastructural data. Together with the initial speed and desired final speed, they form the boundary conditions vector $B(t) \triangleq \{t_f(t), s_f(t), v_i(t), v_f(t)\}$ to the optimization. The process of setting the boundary conditions is described in more detail in Sect. 8.2.6.

Other elements that have to be predicted are the disturbances and constraints acting on the vehicle during the just-defined horizon. A road parameter vector $C(t) \triangleq$

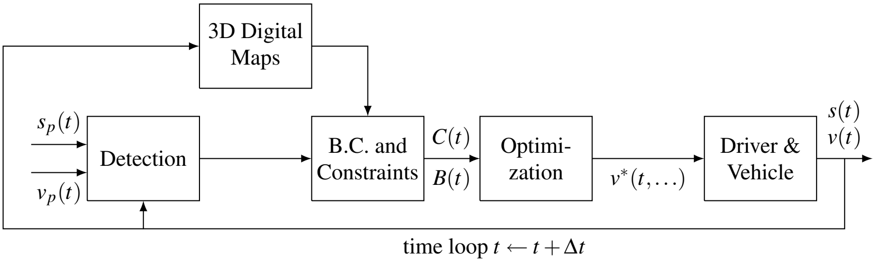


Fig. 8.5 Conceptual sketch of a predictive eco-driving system

$\{\alpha(s(\tau)), R(s(\tau)), v_{max}(\tau), s_p(\tau)\}$, $\tau \in [t, t + t_f(t)]$, can be obtained similarly to what was discussed for the eco-ACC approach.

Once the boundary conditions and the leader and infrastructure constraints are set, an optimization of the speed profile of the host vehicle can be run for the entire horizon. The OCP to be solved is generally that described in Chap. 6, which we rewrite here in a form that is valid for ICEVs and EVs:

$$\begin{aligned}
\underset{\mathbf{u}(t)}{\text{minimize}} \quad & J = \int_{\tau=t}^{t+t_f(t)} P_T(v(\tau), \mathbf{u}(\tau), \alpha(s(\tau)))d\tau \\
\text{subject to} \quad & \dot{s}(\tau) = v(\tau) \\
& \dot{v}(\tau) = f(v(\tau), \mathbf{u}(\tau), \alpha(s(\tau))) \\
& v(t) \triangleq v_i(t), \\
& v(t + t_f(t)) = v_f(t), \\
& s(t) = 0, \\
& s(t + t_f(t)) = s_f(t) \\
& \mathbf{u}_{min}(v(\tau), \tau) \leqslant \mathbf{u} \leqslant \mathbf{u}_{max}(v(\tau), \tau) \\
& 0 \leq v(\tau) \leq \min(v_{lim}(s(\tau)), v_{turn}(R(s(\tau)))) \\
& 0 \leq s(\tau) \leq s_p(\tau)
\end{aligned} \tag{8.3}$$

where $\mathbf{u}$ is the control vector and f is the speed dynamics equation (6.15). For HEVs, the dynamics of the state of charge ξ_b must be additionally taken into account, as well as the additional boundary condition $\xi_b(t + f_f(t)) = \xi_f$. The quantity ξ_f is not known a priori, as it is the result of the SoC trajectory optimization over the whole trip, and not just one sub-trip (see Sect. 6.6). However, it can be approximated by heuristic expressions such as that described by (4.47). Overall, the output of this optimization block is the optimal speed profile for the entire horizon $v^*(\tau)$, $\tau \in [t, t + t_f(t)]$.

Similarly to an eco-ACC system, the optimal speed profile is either advised to the human driver through an HMI (see Sect. 8.3.1) or directly realized through a vehicle controller in the case of an automated speed controller. The actual position and speed of the host vehicle serve as the feedback input to update the boundary conditions for a new iteration after each time step.

8.2 Practical Issues

Many of the building blocks appearing in the flowcharts of the previous sections have been extensively treated in this book. For instance, the use of 3D digital maps is the subject of Sect. 4.1. Also, the characteristics of the various sensors used, not explicitly shown in the flowcharts, can be found in Sect. 3.2.1. However, many corollary functions and sub-systems have not been treated yet with an adequate detail and thus shall be discussed in this section. Among these are, the speed recording and the breakpoint detection of eco-coaching (Sects. 8.2.1 and 8.2.2), the leader position prediction and the MPC algorithms of both eco-ACC and predictive eco-driving (Sects. 8.2.3–8.2.5), as well as predictive boundary condition setting (Sect. 8.2.6).

8.2.1 Speed and Path Recording

As discussed in Sect. 3.2, accurate measurement of vehicle velocity often needs multiple sensors (GPS, IMU, wheel odometer) and the fusion of their output through extended Kalman filtering. Simple and fast-running eco-coaching systems like the one shown in Fig. 8.1 might nevertheless rely only on relatively noisy GPS data beside 3D map data, from which longitudinal velocity (speed), road slope, and road curvature must be extracted. Raw data (latitude, longitude, elevation) on a sub-trip need first to be transformed into local Cartesian coordinates (x_c, y_c, z_c) based on the UTM (Universal Transverse Mercator) system, to which record times t_c, $c = 1, 2, \ldots, C$ are associated. If s is the curvilinear position and θ is the heading angle, numerical differentiation of the data set should in principle reveal the speed,

$$v(s) = \frac{ds}{dt} \ , \tag{8.4}$$

the curvature

$$\frac{1}{R(s)} = \frac{d\theta(s)}{ds} \ , \tag{8.5}$$

and the slope

$$\alpha(s) = \arctan\left(\frac{dz(s)}{ds}\right) \ , \tag{8.6}$$

with

$$dx = \cos\theta ds \ , \tag{8.7}$$

$$dy = \sin\theta ds \ . \tag{8.8}$$

To differentiate the available data numerically, a smoothing filter such as the *Savitzky–Golay filter* [3] can be used given its ability to increase the signal-to-noise ratio without greatly distorting the signal [4]. With this approach, the curvature can be obtained directly from the first- and second-order position derivatives, without the intermediate evaluation of the heading.

Another approach consists of setting an ad-hoc optimal control problem to find the time profiles of speed, curvature, and slope that minimize the error between GPS data (x_c, y_c, z_c) and reconstructed positions evaluated at times t_c [5]. To do so, time is chosen as the independent variable,[3] while the "control" vector consists of the derivatives of the sought functions $\tilde{u}(t) \triangleq [dv/dt, d(1/R)/dt, d\alpha/dt]$. The performance index to be minimized can be defined for the i-th sub-trip as

$$\int_0^{t_f} r\tilde{u}^2 dt + \sum_{c=1}^{C} \left((x_c - x(t_c))^2 + (y_c - y(t_c))^2 + (z_c - z(t_c))^2\right) \ , \qquad (8.9)$$

with (8.4)–(8.8) as dynamic equations and t_f given by the first of (8.1). The weighting factor r trades off the variations in the reconstructed curvature profile with the reconstruction accuracy. Appropriate boundary conditions and constraints complete the OCP.

More accurate estimation might be obtained using IMU data and/or sensor fusion, particularly for on-board systems that are integrated with the in-vehicle data bus.

8.2.2 Breakpoint Detection

In analyzing a recorded speed and path profile, breakpoints can be defined with two types of considerations. On one hand, fixed or stationary breakpoints can be defined at locations where: (i) slope α, (ii) road curvature R, or (iii) top speed limit v_{lim} change more than a predefined threshold with respect to their value in the previous sub-trip, or (iv) the trip is likely to undergo a discontinuity, like in the presence of signalized intersections, stops, or other types of intersections.

On the other hand, the analysis of the actual speed profile $\hat{v}$ generally reveals additional breakpoints that were likely induced by surrounding traffic, which is not predictable only using static road information. The detection of such breakpoints can be done, in principle, by identifying the speed minima (including stops) subsequent to decelerations that are not already accounted for by the static breakpoints.

Minima detection is, however, not always a simple task as the recorded speed profile can be very noisy and spurious minima can appear. The latter could be eliminated by keeping, e.g., only minima that have a sufficient prominence[4] or that have a minimum time separation from other minima. These features can be imposed while using functions that are available in commercial software, for example Matlab's *findpeak*.

However, often it is not sufficient to isolate the representative breakpoints. Digital filters such as the aforementioned Savitzki–Golay filter [3] can smooth the data and reduce the noise, but they are an option only if they preserve the main minima that can be attributed to breakpoints. Ramer–Douglas–Peucker algorithm [6] can be used

[3] Curvilinear position is the independent variable in the original formulation of [5], although this choice generates some difficulties.

[4] Analogously to the prominence of a peak, the prominence of a valley measures how much the valley stands out due to its intrinsic depth and its location relative to other valleys. A shallow isolated valley can be more prominent than one that is deeper but is an otherwise unremarkable member of a deep sequence.

to decrease the number of points in the recorded speed trace, which can also reduce the noise. Mathematical morphology can also be used, directly on the speed trace or rather on a filtered signal, to isolate the valleys from the original signal. However, the position of these valleys might be displaced when their shape is strongly asymmetric.

8.2.3 Leader Position Prediction

The optimality of the iterative schemes of Figs. 8.4 and 8.5 is strongly affected by the quality of prediction of the preceding vehicle's states. The simplest approach to predicting the leader's future speed is to assume that the vehicle continues to move with the same acceleration until it stops or exceeds a maximum velocity [7]. But this method can be unrealistic because it may lead to a very high or a zero predicted velocity at the end of a long horizon. Therefore, this model is often [8] improved by considering a speed-dependent acceleration value.

To this end, the car-following models described in Sect. 4.2.1 could in principle be applied to the leader and the generic equation (4.2) integrated over the prediction horizon to yield $v_p(\tau)$, $\tau \in [t, t + t_f(t)]$. Normally, the information about the speed of the vehicle preceding the leader and its gap from the leader are not available, so that (4.2) is integrated assuming either free flow,

$$\dot{v}_p(\tau) = F(v_p(\tau), \delta \to \infty) \ , \tag{8.10}$$

or approaching a fixed obstacle, e.g., a red light or an intersection, whose position is known,

$$\dot{v}_p(\tau) = F(v_p(\tau), \delta(\tau)) \tag{8.11}$$

$$\dot{\delta}(\tau) = -v_p(\tau) \ . \tag{8.12}$$

The integration of these differential equations in closed form is only possible for very simple car following models. For more realistic ones, such as the Gipps' model or the IDM, numerical integration is required, which is not always practical in an onboard system.

Therefore, more heuristic approaches have been often used in the literature. For example, introducing a multiplier to weigh the current acceleration as

$$a_p(\tau) = a_p(t) \cdot \zeta(v_p(\tau)) \ , \tag{8.13}$$

where $\zeta(v_p)$ is defined as

$$\zeta(v_p) \triangleq \frac{1}{(1 + e^{-\beta_1(v_p - w_1)})(1 + e^{\beta_2(v_p - w_2)})} \ . \tag{8.14}$$

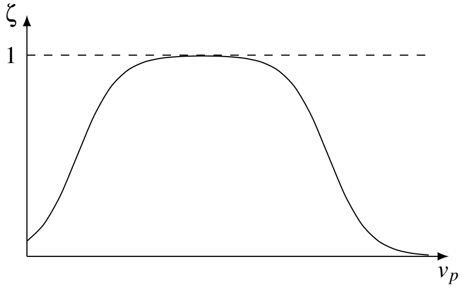


Fig. 8.6 Function (8.14)

The parameters w_1 and w_2 define an approximate range of velocities. The function (8.14) states that $a_p(\tau) \approx a_p(t)$ if $v_p(\tau)$ is within that range, otherwise $a_p(\tau) \approx 0$, and $\beta_1 > 0$ and $\beta_2 > 0$ express the sharpness of the sigmoid function, see Fig. 8.6. That means that the acceleration of the preceding vehicle approaches zero when it reaches a maximum velocity or stops completely.

In the case the preceding vehicle is approaching a red signal, prediction using (8.14) is not appropriate, since it is certain that the vehicle will stop at the end of the section after a known distance $\delta(t) = s_f - s_p(t)$. For this special case, a prediction model of the preceding vehicle based on experimental driving data has been proposed in [7] as

$$v_p(\delta(\tau)) = \sum_{i=0}^{5} c_i \delta^i(\tau) = -\dot{\delta}(\tau) \ , \tag{8.15}$$

with the coefficients c_i experimentally determined. The corresponding stopping rate is obtained as $a_p(\tau) = -v_p(\delta(\tau)) \cdot dv_p(\delta)/d\delta$.

With V2V connectivity, the future intent of the preceding vehicle can be communicated to the follower, which reduces the uncertainty. A vehicle cannot know its future moves with certainty but if it employs an MPC approach for longitudinal control it generates, as a by product, a future sequence of its optimal control moves conditioned on its current state. This "intended" sequence of its future moves can be communicated to following vehicles as a best estimate of what it is going to do. The ego following vehicle can replace, in its cost function and constraints, the intended position of the preceding vehicle over the horizon. The MPC problems can be solved sequentially by each participating CAV from the front to the tail of a string resulting in a distributed MPC solution of the car following problem as also described in [9].

Even in absence of V2V connectivity, latest sensor observations can be used to estimate the probable position of the preceding vehicle along a future planning horizon. In [10–12], different flavors of a Markov chain approach are proposed to predict the likely position of the preceding vehicle over ego car planning horizon. By counting historical occurrences, probabilities are assigned to the transition between two states (i.e., vehicle velocity) in consecutive time steps. In a Markov chain model the assumption is that a state at any point in time is only dependent on the states of one prior step and not earlier steps. In [11] only velocity transitions of the preceding vehicle were modeled, but more precise predictions can be made if enough data is available to include the acceleration state as well. To reduce the size of the transition probability matrix[5] and the need for training data it is possible to use only a categorical variable describing whether the preceding vehicle is accelerating, decelerating, or cruising. For instance, the velocity can be quantized at 1 m/s steps between 0 and maximum allowable velocity while the acceleration state can be chosen from {accelerating, decelerating, cruising} as shown in [12]. A higher resolution quantization of acceleration is used in [13, 14].

Once the velocity of preceding vehicle is estimated and given the current position of the preceding vehicle $s_p(0)$, future positions can be determined using kinematics. Because a model is used to generate the final output, this approach always gives physically realistic trajectories. To each sequence of velocities and resulting positions a probability calculated from the constructed Markov transition matrix can be assigned. With a sufficient number of samples, a distribution can be fit to the generated position histogram to be used in anticipative car following as described in a case study in Sect. 9.3.

8.2.4 Probabilistic Traffic Light Prediction

There can be much uncertainty in the phase and timing of a traffic signal which makes predicting its future state quite challenging. For fixed-time traffic signals which do not respond to traffic conditions (see Sect. 4.1.3) and operate only on a timing table, the traffic signal clock may drift during a 24 h period (this is potentially due to variations in the electric grid frequency, see [15] for more details). Therefore, it is not possible to know with certainty the start of greens and reds, even for fixed-time signals. The level of uncertainty is higher for actuated and adaptive traffic signals which do respond to traffic conditions. Although they have a base timing table, the timings of actuated and adaptive lights may change according to traffic conditions, rendering not only the start of reds and greens but also the phase lengths uncertain.

Due to the aforementioned uncertainties, it is difficult to always determine the start and duration of greens deterministically. One can employ a probabilistic prediction framework to handle the case with partial or uncertain information. Here we describe the approach that first appeared in [16].

Assume that the current phase (color) and the average red and green lengths for a signal are known. This information can be used to predict the probability of a green over the planning horizon. The state of a light is denoted by $\ell(t)$ which can assume two values, g and r, representing green and red respectively. The goal is to determine the probability of a light being green at time $t + t_p$ conditioned on its current color at time t. To form this conditional probability function, the durations of green and red are assumed to be t_g and t_r respectively on average. The traffic signal is assumed

[5] A matrix $P_{\alpha\beta} \triangleq P\left(v_p(\tau + \Delta\tau) = \beta | v_p(\tau) = \alpha\right)$ indicating the probability to go from velocity state α to state β.

to operate cyclically, which is true for many traffic signals, and as a result the total cycle time is fixed and equal to $t_g + t_r$. Here yellow time is included in red.

Using relatively straight-forward probabilistic reasoning, the chance of a green light in t_p seconds, given a green at current time t can be found to be:

$$P[\ell(t+t_p) = g|\ell(t) = g] = \begin{cases} \dfrac{t_g - t_m}{t_g}, & \text{if } t_m \le t_r \wedge t_m \le t_g \\ \dfrac{t_g - t_r}{t_g}, & \text{if } t_r \le t_m \le t_g \\ 0, & \text{if } t_g \le t_m \le t_r \\ \dfrac{t_m - t_r}{t_g}, & \text{if } t_g \le t_m \wedge t_r \le t_m \end{cases}, \tag{8.16}$$

where $t_m \triangleq \operatorname{mod}(t_p, t_g + t_r)$ is the residue of division of t_p by $t_g + t_r$ and its role is to remove whole cycle times from t_p. In other words because the signal clock is assumed to be periodic, the resulting conditional probability is also going to be a periodic function of time so the probability of green (or red) can be described in terms of t_m instead of t_p.

Now for instance when $t_m \le \min(t_g, t_r)$, then $\ell(t + t_m) = g$ for $0 \le t < t_g - t_m$ and $\ell(t + t_m) = r$ for $t_g - t_m \le t \le t_g$, so that the probability of light being green now and in t_m seconds is $\frac{t_g - t_m}{t_g}$, which explains the first line of Eq. 8.16. But if $t_g \le t_m \le t_r$ and the light is green now, it is impossible for it to be green in t_m seconds as in third line of Eq. 8.16. The details may be best understood graphically and therefore they are not explained here.

Similarly, the chance of a green light in t_p seconds, given a red at time t is:

$$P[\ell(t+t_p) = g|\ell(t) = r] = \begin{cases} \frac{t_m}{t_r}, & \text{if } t_m \le t_r \wedge t_m \le t_g \\ 1, & \text{if } t_r \le t_m \le t_g \\ \frac{t_g}{t_r}, & \text{if } t_g \le t_m \le t_r \\ \frac{t_g + t_r - t_m}{t_r}, & \text{if } t_g \le t_m \wedge t_r \le t_m \end{cases}. \tag{8.17}$$

Figures 8.7 and 8.8 show several probabilistic prediction examples with different splits between red and green but with the same cycle length. These are visualizations of the probabilities which we use in probabilistic simulation cases described later in Sect. 9.1.

8.2.5 MPC Schemes

In contrast to the *offline* optimal control framework described in Sects. 6 and 7, MPC needs to solve an optimization problem *online* at each control time step to compute the optimal control input. This fact has prevented the application of MPC in several

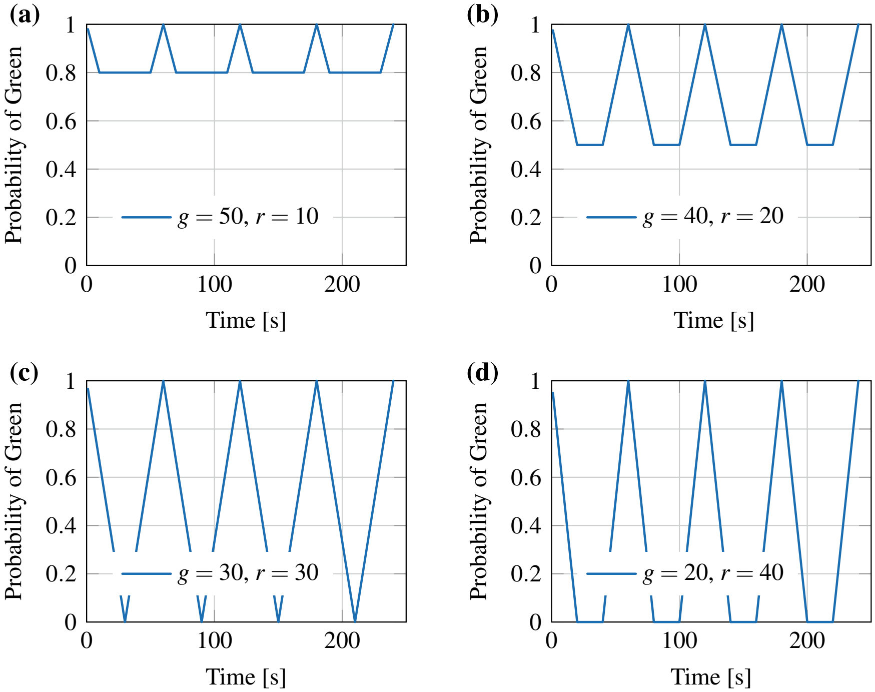


Fig. 8.7 Conditional future probability of green given that the light is currently green, for four different light timing patterns. In all patterns the total cycle time is 60 s, with the lengths of green and red indicated in the legends. The time axis is t_p as described in (8.16)–(8.17)

contexts, either because the processor technology needed to solve the OCP within the sampling time is too expensive or simply infeasible for onboard automotive-grade controllers, or because the implementation of the numerical solver causes software certification and/or explicability concerns, especially in safety-critical applications. At the same time, such limitations have motivated the development of *explicit MPC* approaches, where the optimal control inputs are pre-computed offline as a function of the states [17], or even the analytical and semi-analytical approaches presented throughout this book. However, in non-critical and research applications, numerical MPC remains the most common option.

Practical methods to solve problem (8.3) online generally require its discretization, that is, replacing the integral J with a sum

$$\sum_{\tau=t}^{t+t_f(t)} P_T(\cdot)\Delta\tau \tag{8.18}$$

and the dynamic equations with algebraic equations $v(\tau + \Delta\tau) = v(\tau) + f(\cdot)\Delta\tau$, $s(\tau + \Delta\tau) = s(\tau) + v(\tau)\Delta\tau$, with the prediction time step $\Delta\tau$ not necessarily equal

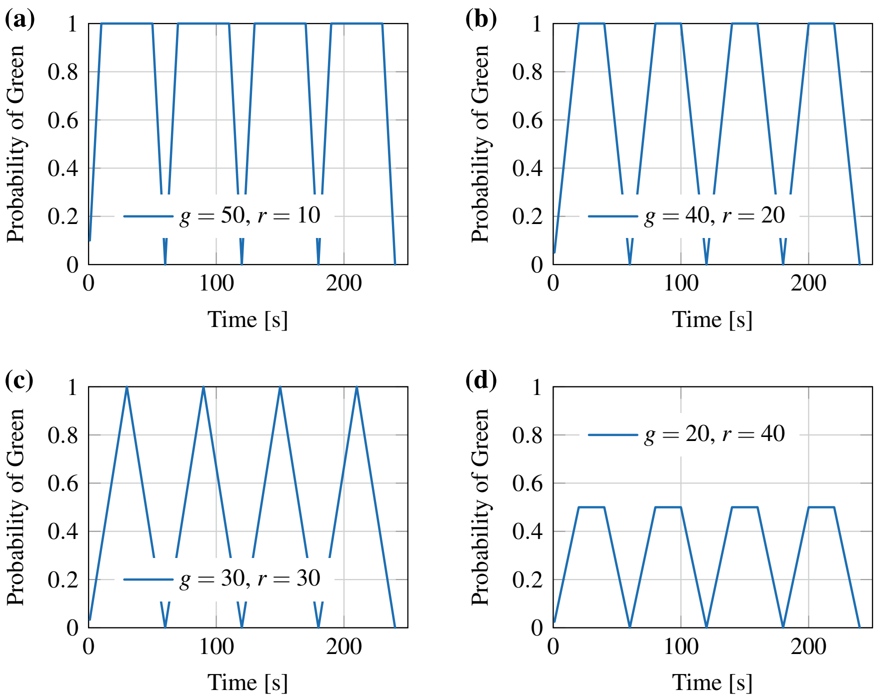


Fig. 8.8 Conditional future probability of green given that the light is currently red, for four different light timing patterns. In all patterns the total cycle time is 60 s, with the lengths of green and red indicated in the legends. The time axis is t_p as described in Eqs. 8.16 and 8.17

to the control time step Δt. The stability of MPC has been proven at least for OCP without terminal state constraints and terminal cost [18]. An excellent survey on stability in MPC is given in [19].

For linear systems with linear constraints and a quadratic cost, the discretized problem can be rewritten into a standard quadratic programming (QP) form, as it is the case, for instance, of the quadratic EV model (6.74). A variety of methods (gradient, interior-point, etc.) are commonly used to solve QP problems exactly and many fast and efficient solvers are available, including toolboxes in the main programming languages [20].

More generally, nonlinear MPC (NMPC) problems must be solved iteratively and involve approximations of the problem equations. As far as optimal control is concerned, the two solution approaches can be recognized, those of indirect and direct methods.

Indirect methods, based on the PMP (see Sect. 6.2.2) and resulting in a two-point boundary value problem, suffer from a general lack of efficient and robust generic software packages to be implemented online. In contrast, the use of *direct methods* is nowadays much more common. With this approach, the NPMC must be converted into a nonlinear programming (NLP) problem. To do that, pseudospectral (orthogonal collocation) algorithms are commonly employed, which use polynomials to approximate states and control inputs at some collocation points. Depending on the choice of the polynomial basis and of the collocation points, several pseudospectral methods are available, such as those named after Gauss (GPM), Chebyshev (CPM), and Legendre (LPM).

Once cast into an NLP, the latter can be solved using the several existing Newton-type methods. These are roughly classified into sequential quadratic programming (SQP) and interior-point (IP) methods [21, 22]. For both categories, several solvers and software packages are available [22]. Energy-efficient driving systems have been actually implemented with SQP solvers such as NPSOL [23] and SNOPT [24], as well as with an IP-type solver such as C/GMRES [25] and IPOPT.

The use of such methods needs, however, special treatments of the original OCP to, e.g., regularize constrained or singular arcs, smooth discontinuous functions, etc. The use, when available, of intrinsically robust, analytical or semi-analytical solution procedures, such as those presented in this book, could overcome these difficulties.

8.2.6 Setting of Boundary Conditions

As discussed in Sect. 8.1.4, the optimization block in the predictive eco-driving scheme of Fig. 8.5 requires proper boundary conditions

$$B(t) = \{t_f(t), s_f(t), v_i(t), v_f(t)\} \ , \tag{8.19}$$

where the argument t explicitly denotes that such setting is made at each new optimization run.

The initial speed of the new optimization horizon is naturally set to the current speed,

$$v_i(t) = v(t) \ .$$

As for the other boundary conditions, their definition results from the knowledge of the road network ahead and the estimated leader position. We shall assume good predictive capability of both these elements, which form the road parameter vector $C(t) = \{\alpha(s(\tau)), v_{max}(\tau), s_p(\tau)\}$, $\tau \in [t, t + t_f(t)]$.

The spatial horizon can be either constant or variable. The former option, $s_f(t) \equiv s_f$, is also known as *sliding horizon* and is sometimes appropriate for highway trips, in the absence of any sensible breakpoints. This is the preferred option for PCC and eco-ACC implementations, as discussed in Sects. 8.1.2 and 8.1.3.

In most driving situations, however, road topology naturally fixes the next breakpoint. Referring to the scenarios of Sect. 7, at least four situations may occur, where the next breakpoint is the position (i) where road slope and/or curvature change sensibly with respect to the current value, (ii) where a different speed limit than the current is posted, (iii) of a signalized intersection, or (iv) of a stop. Denoting the position of the next breakpoint as $\hat{s}_{NB}(t)$, where the hat denotes an estimation and the argument t the fact that such estimation can vary with time during the trip, the spatial horizon is naturally set as

$$s_f(t) = \hat{s}_{NB}(t) - s(t) \; . \tag{8.20}$$

As for the temporal horizon, it also is rarely fixed and sliding. More often, it is estimated based on $\hat{s}_{NB}$ and some desired or average speed (e.g., traffic-induced) $V(t)$ that is needed to reach it,

$$t_f(t) = \frac{s_f(t)}{V(t)} \; . \tag{8.21}$$

Though, in case (iii) of a signalized intersection, it is more natural to set a breakpoint time $\hat{t}_{NB}(t)$ during the green phase and, similarly to (8.20), evaluate the time horizon as the remaining time to catch the green,

$$t_f(t) = \hat{t}_{NB}(t) - t \; . \tag{8.22}$$

The terminal speed $v_f(t)$ also is to be estimated. In case (i) above, that is a typical highway situation, often $v_f(t) = v_i$ (cruising). In case (ii) to avoid having to abruptly decelerate or brake at the beginning of the new segment, the obvious choice is to set $v_f = \hat{v}_{max,NB}(t) - \Delta v$, where $\hat{v}_{max,NB}$ is the speed limit in the next sub-trip and Δv is a safety margin to allow further acceleration in the next sub-trip. In case (iii), $v_f(t)$ should equal the crossing speed v_t at the traffic light, and could in principle be the result of a further optimization as shown in Sect. 7.7.[6] In case (iv), $v_f(t) \equiv 0$.

The feasibility of the sub-trip horizons defined in one of the ways discussed above must be checked. For that, the feasible range of durations and distances satisfying all constraints, $F(t_f(t), s_f(t)) \geq 0$, must be constructed as shown in Chap. 7 using the available information contained in the road parameter vector $C(t)$. Additionally, the vehicle and powertrain characteristics can be used to set the maximum acceleration or deceleration.

If the pair $(s_f(t), t_f(t))$ is not feasible, a correction must occur. Two cases may be identified. In case, e.g., of a preceding vehicle that is slower than thought ($V(t)$ overestimated in (8.21)), then likely an increase of $t_f(t)$ suffices to bring the boundary conditions back in the feasible range. If, however, a non-estimated obstacle appears, for instance, the preceding vehicle will stop before the estimated position $\hat{s}_{NB}$, then the most natural option is to decrease $s_f(t)$ and possibly $t_f(t)$ according to (8.21). These two situations are illustrated in Fig. 8.9. In both cases, a safety margin can be conveniently applied with respect to the closest boundary of the feasibility domain.

[6] In this OCP, $v_f(t)$ is open so the corresponding costate is fixed.

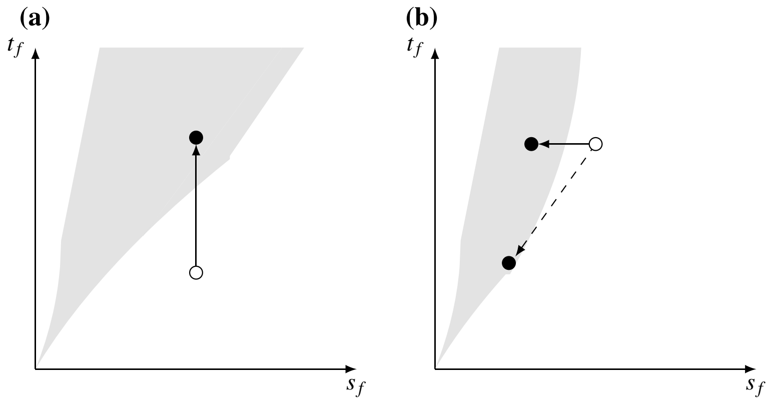


Fig. 8.9 Correction of boundary conditions that are infeasible for an underestimation of t_f (left) and an overestimation of s_f (right)

8.3 On-Board Implementation

The effective implementation of the concepts presented in the previous sections requires several technical arrangements. Here we discuss the ergonomy of the HMIs aimed at advising drivers about eco-practices (Sect. 8.3.1) and of their response to these advice (Sect. 8.3.2). Finally some issues related to the automated implementation are discussed in Sect. 8.3.3.

8.3.1 Human-Machine Interfaces

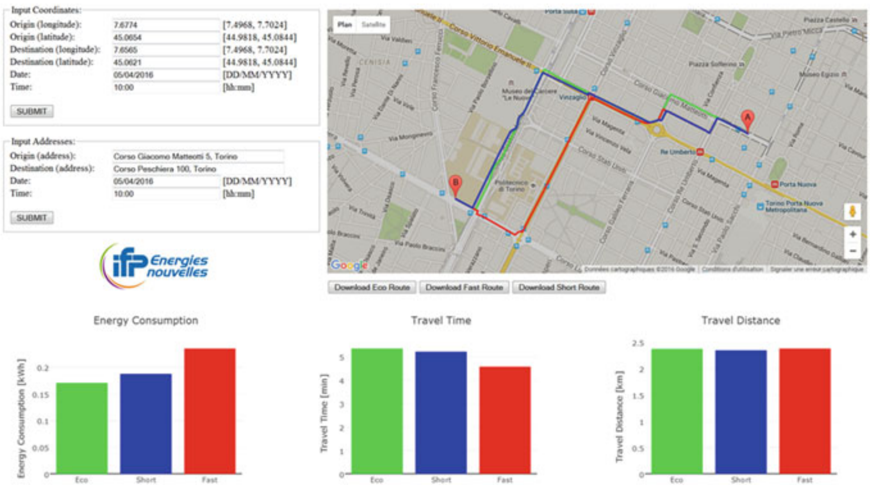
The energy-efficiency systems described in the previous sections and chapters must be complemented with human-machine interfaces to communicate with a human driver. As discussed above, the role of an eco-coaching HMI is to effectively assess the driver's behavior in-trip or post-trip; that of an eco-driving HMI is to advise a speed to follow, and possibly other actions. In addition, eco-routing HMIs advise the route to follow.

The main types of driver assistance systems are: (i) visual interfaces, (ii) visual-auditory assistance systems, and (iii) haptic assistance systems.

8.3.1.1 Visual and Acoustic Assistance Systems

Visual and visual–acoustic assistance systems are based on dedicated on-board multimodal displays or personal mobile devices (smartphones, tablets). The system architecture may include several tabs, related to different functions or different drive stages (e.g., pre-trip eco-routing, in-trip eco-driving, post-trip eco-coaching).

Eco-routing output is usually presented similarly to current navigation systems, with the user providing origin and destination by entering physical addresses or



Input Coordinates:

Origin (longitude):	7.6774	[7.4968, 7.7024]
Origin (latitude):	45.0654	[44.9818, 45.0844]
Destination (longitude):	7.6565	[7.4968, 7.7024]
Destination (latitude):	45.0621	[44.9818, 45.0844]
Date:	05/04/2016	[DD/MM/YYYY]
Time:	10:00	[hh:mm]

Input Addresses:

Origin (address):	Corso Giacomo Matteotti 5, Torino	
Destination (address):	Corso Peschiera 100, Torino	
Date:	05/04/2016	[DD/MM/YYYY]
Time:	10:00	[hh:mm]

Fig. 8.10 Example screenshot of an eco-routing interface (http://isntsv-optem:8080/EcoRouting/), showing the user's inputs addresses and coordinates, the routes displayed on a map, and the corresponding energy, time, and distance attributes

clicking on a map, and possibly time of departure (if not automatically detected). The system then displays the computed eco-route on the map. Information on predicted trip energy consumption, duration, and distance as compared with standard route choices (typically, the fastest and the shortest route) can be also useful for the driver. A screenshot of an example eco-routing interface is shown in Fig. 8.10.

In-trip or post-trip eco-coaching HMIs must provide the user with useful feedback on his/her driving style. For this purpose, scores are commonly used. These scores can be indirect or direct. An example of the first type is the time fraction spent with a vehicle speed different from the optimal. On the other hand, direct scores compare the actual (tank) energy consumed[7] during the sub-trip, E_T, with the calculated optimum, E_T^*, for instance using the definition

$$\text{EDS} \triangleq 10\left(2 - \frac{E_T}{E_T^*}\right) . \tag{8.23}$$

The numerical coefficients of (8.23) ensure that, when $E_T = E_T^*$, the score is equal to 10 while, if the actual consumption is twice the minimum, then the score is zero.

Scores can be presented by exact values, horizontal bars, or various *visual metaphors*, see the screenshots in Fig. 8.11. A history with scores for recent sub-trips or entire trips is also often useful, particularly if the system allows calculating analytics, e.g., the average score of all the saved trips. That allows the driver to compare his/her current performance with recent ones and to get an impression about

[7] Note that the means to actually measure the energy consumption on board is a complex subject that is beyond the scope of this book.

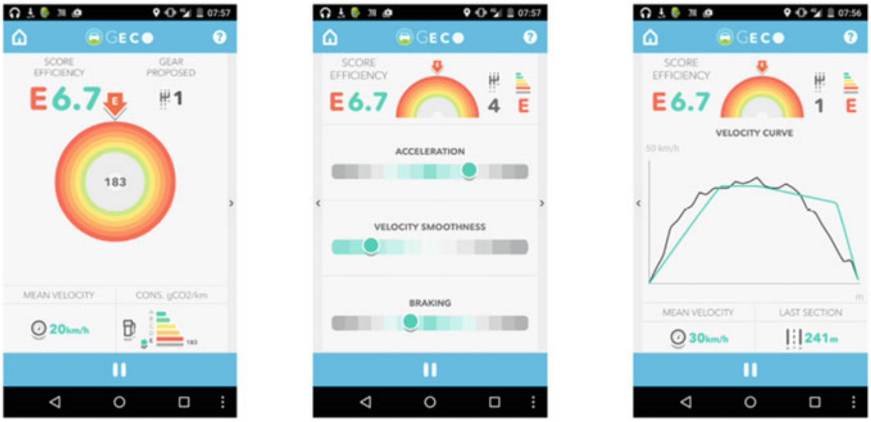


Fig. 8.11 Example screenshots of an eco-coaching interface (GECO [26]) showing a presentation of the EDS scoring as a numerical value and of the energy consumption (in CO_2 g/km) as a metaphor (left); three sliders rating accelerations, decelerations, and smoothness (middle); a visual comparison of the optimal vs. the actual velocity profiles in the last sub-trip (right)

his/her progress. This learning process supports the long-term motivation for an energy-efficient behavior of the driver.

Since they must induce a desired behavior of the driver, predictive eco-driving HMIs might be more complex. They typically include several kinds of icons, representative of different events during the drive, visual advices, feedback on fulfillment of the eco-driving behavior through scores, as well as game (intrinsic reward) elements [27].

Visual advice can be provided by presenting the current speed calculated by the eco-driving function as a numerical value, on a tachometer-like icon, or as a horizontal bar in comparison with the current actual speed. Moreover, the recommended speed can be conveniently visualized as a range, besides its exact computed value [28]. Second-order information on the required change in pedal depression has been demonstrated to be more effective than first-order information on the current speed error [29].

In addition to visual advise, feedforward information of upcoming events (traffic light with distance and possibly status, traffic sign, curves, roundabout, etc.) is often also presented in eco-driving assistance interfaces [27].[8]

The use of game elements in the driver assistance context is also increasingly popular [30]. The objective of such gamified interfaces is to facilitate intrinsically-motivated energy-saving behaviors that are less amenable to change than with simple prescriptions. According to the *self-determination theory*, a psychology theory of human personality and motivation, the factors to be considered when developing such interfaces include: (i) avoiding external rewards (e.g., prizes, cash incentives);

[8] See a few examples in the case studies of Chap. 9.

(ii) providing positive feedback to drivers regarding their eco-driving behavior; and, (iii) fostering social interactions (e.g., through cooperative systems) [30].

Game elements can be provided to drivers through challenges based on different levels of difficulty, to facilitate feelings of enhanced competence as they progress through levels. Research in this field has highlighted the need to balance difficulty and user skill, in order to avoid boredom (when challenges are too easy) and/or frustration (when challenges are too difficult).

Social-interaction features can be designed to let the users share their scores with friends or other drivers, in order to socially engage and cooperate toward the common goal of eco-driving. Competitive challenges for users or teams (of friends, co-workers) are also organized by providers of eco-driving systems and local authorities to promote the adoption of energy-saving driving styles [31].

8.3.1.2 Haptic Assistance Systems

In a typical vehicle, there is a proportional relationship between the force applied to the accelerator pedal and the pedal depression produced. In a haptic system, this relationship is changed by increasing or decreasing the accelerator pedal resistance. The variable force-depression profile should somehow reflect the acceleration that is advised to the drivers and thus expected from them. Two methods have been demonstrated in recent years [29], see Fig. 8.12.

The *haptic force method* requires the driver to produce a significant extra force to increase the pedal depression beyond that calculated by the eco-driving function to be optimum for energy efficiency at the current instant. In contrast, to discourage under-accelerating, a weakening of the pedal resistance below that of a standard accelerator pedal is enforced for pedal depressions below the advised one, so that the pedal is easier to push than in a vehicle not equipped with the haptic assistance system.

The *haptic stiffness method* discourages an over-acceleration through a change in pedal stiffness rather than a step-change in force for the drivers to overcome. Again, this method is able to encourage harsher acceleration when energy efficiency can be improved by increasing pedal force, through a reduction in the resistance of the accelerator pedal relative to a non-haptic standard pedal.

The effectiveness of these two methods and in comparison to visual assistance systems is still a matter of debate, with the few experimental results being somehow contradictory [29].

8.3.2 Human Response

The effectiveness of energy-efficient driving assistance systems depends not only on the algorithms and the HMI used, but also strongly on how the human drivers react to the advice and adopt it.

Effectiveness of the system as a whole can be evaluated through field tests, driving simulators, or traffic simulations. Commonly, multiple participants are selected for

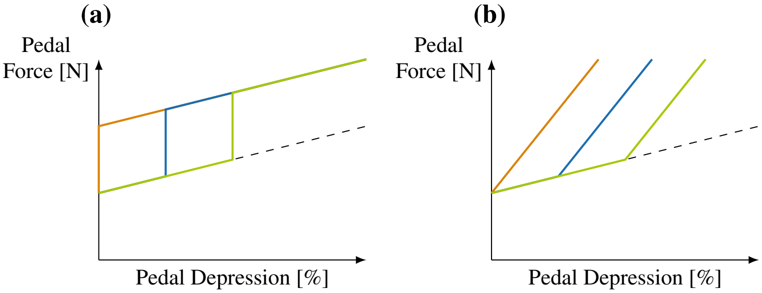


Fig. 8.12 Haptic force (**a**) and haptic stiffness (**b**) system profiles for three setpoints of pedal depression. Dashed: standard pedal; color curves: increasing acceleration setpoints

such tests, and the same driving scenarios are completed at several separate occasions. To limit the influence of external factors, such repetitions are aiming at performing or reproducing, e.g., conditions at the same day of the week and departure time of the day. Though often professional drivers are involved in such tests for practical reasons, the use of non-instructed participants should be favored.

The outcomes of these tests can be of a different nature. In particular, the human response to HMIs can be assessed by several measures. Questionnaires filled by participants to an experimental campaign may inform about the user perception of the interface design (how engaging it is, how easily it is understood, its perceived usability), their self-perception while using the system (e.g., self-evaluating their level of anxiety, annoyance, curiosity), or about the impact of using the interface on driving.

Quantitative indexes are also used in the ergonomic science, such as the percentage of success in correctly reacting and the reaction time to feedback. To evaluate the level of distraction induced by the use of the HMI, the duration of driver's eye glances to the interface(s) and their relative occurrence with respect to those directed to the road scene ahead, are often used [32].

Of course the energy actually spent is the ultimate measure of effectiveness of the system as a whole [33]. The energy efficiency benefits of using an eco-driving system should be evaluated both in the short and in the long term, where the superiority over classical eco-driving trainings should be more apparent.

8.3.3 Automated Drive

By removing the burden from human drivers, automated drive systems are expected to significantly ease implementation of eco-driving functions. This could be as simple as issuing the eco-speed or -lane determined at the motion planning layer (Sect. 3.3.3) to the lower level feedforward and feedback motion controllers (Sect. 3.3.4) that actuate the throttle, the brake, and the steering control motors.

For instance in anticipative car following, an MPC planner issues its acceleration command to the low level controllers. In one possible implementation, pre-mapped feedforward controllers could map the commanded acceleration to throttle or brake pedal positions, while a feedback loop is in charge of velocity tracking. A potential challenge is mismatch between simplified models used at the planning layer and actual dynamics of the vehicle, which could cause degradation in tracking performance. One possible solution in an MPC framework is augmenting a (step) disturbance observer [34] to the simplified model thus introducing integrating modes that improve tracking performance. Computation and communication delays need to be accounted for and addressed or they may cause poor tracking or chattering.

There are challenges at the low-level control layers as well. Poorly designed low-level controllers can diminish the energy gains planned at higher level or could even cause an increase in energy use. For example, an over-aggressive low-level controller may cause chatter between throttling and braking in an effort to precisely follow the eco-speed. This could arise for instance in a vehicle with an internal combustion engine if the response time of the engine is not properly taken into account. The situation can be worse if the vehicle is having a turbocharger or an automatic transmission with torque converter, both of which add extra delays to the vehicle response. Controller de-tuning may be an option in scenarios such as eco-approach to traffic signals but may not be possible for instance in tight platooning. Electric vehicles may be easier to control in similar scenarios due to their more responsive electric motors and drivetrain structures.

Control and perception modules can operate at their own pace and communicate via an environment such as Robot Operating System (ROS) using publish and subscribe functions [35]. A highly sophisticated motion planner that incorporates macro traffic planning may run on a backend cloud. In this case, the communication between the motion planner and the vehicle modules can be achieved by communication networks such as dedicated short-range communications (DSRC) and cellular communications. In all these circumstances, global time-stamping of data would be preferable for synchronization. In addition, potential risks such as deadlocks, time delays, and data losses should also be considered in the system designs and implementations.

References

1. Dib Wissam, Chasse Alexandre, Moulin Philippe, Sciarretta Antonio, Corde Gilles (2014) Optimal energy management for an electric vehicle in eco-driving applications. Control Eng Pract 29:299–307
2. Thibault L, De Nunzio G, Sciarretta A (2018) A unified approach for electric vehicles range maximization via eco-routing, eco-driving, and energy consumption prediction. IEEE Trans Intell Veh 3(4):463–475
3. Savitzky A, Golay MJE (1964) Smoothing and differentiation of data by simplified least squares procedures. Anal Chem 36(8):1627–1639

4. Ojeda LL, Chasse A, Goussault R (2017) Fuel consumption prediction for heavy-duty vehicles using digital maps. In: Proceedings of international conference on intelligent transportation systems (ITSC), pp 1–7. IEEE
5. Perantoni G, Limebeer DJN (2014) Optimal control for a formula one car with variable parameters. Veh Syst Dyn 52(5):653–678
6. Douglas DH, Peucker TK (1973) Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartogr: Int J Geogr Inf Geovisualization 10(2):112–122
7. Kamal MAS, Mukai M, Murata J, Kawabe T (2011) Ecological driving based on preceding vehicle prediction using MPC. IFAC Proc Vol 44(1):3843–3848
8. Kamal MAS, Mukai M, Kawabe T (2013) Model predictive control for ecological vehicle synchronized driving considering varying aerodynamic drag and road shape information. SICE J Control Meas Syst Integr 6(5):299–308
9. Zheng Y, Li SE, Li K, Borrelli F, Hedrick JK (2017) Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies. IEEE Trans Control Syst Technol 25(3):899–910 May
10. McDonough K, Kolmanovsky I, Filev D, Szwabowski S, Yanakiev D, Michelini J (204) Stochastic fuel efficient optimal control of vehicle speed. In: Optimization and optimal control in automotive systems, pp 147–162. Springer
11. Zhang C, Vahidi A (2011) Predictive cruise control with probabilistic constraints for eco driving. In: Proceedings of dynamic systems and control conference and symposium on fluid power and motion control, pp 233–238. American Society of Mechanical Engineers
12. Wan N, Zhang C, Vahidi A (2019) Probabilistic anticipation and control in autonomous car following. IEEE Trans Control Syst Technol 27:30–38
13. Dollar RA, Vahidi A (2017) Quantifying the impact of limited information and control robustness on connected automated platoons. In: Proceedings of international conference on intelligent transportation systems (ITSC), pp 1–7. IEEE
14. Dollar RA, Vahidi A (2018) Efficient and collision-free anticipative cruise control in randomly mixed strings. IEEE Trans Intell Veh 3:439–452
15. Lin P-S, Fabregas A, Chen H, Rai S (2010) Impact of detection and communication degradations on traffic signal operations. In: Proceedings of annual meeting and exhibit, ITE
16. Mahler G, Vahidi A (2014) An optimal velocity-planning scheme for vehicle energy efficiency through probabilistic prediction of traffic-signal timing. IEEE Trans Intell Transp Syst 15(6):2516–2523
17. Alessio A, Bemporad A (2009) A survey on explicit model predictive control. In: Nonlinear model predictive control, pp 345–369. Springer
18. Zanon M, Frasch JV, Vukov M, Sager S, Diehl M (2014) Model predictive control of autonomous vehicles. In: Optimization and optimal control in automotive systems, pp 41–57. Springer
19. Mayne DQ, Rawlings JB, Rao CV, Scokaert POM (2000) Constrained model predictive control: stability and optimality. Automatica 36(6):789–814
20. Gould NIM, Toint PL (2000) A quadratic programming bibliography. Numer Anal Group Intern Rep 1:32
21. Betts JT (2010) Practical methods for optimal control and estimation using nonlinear programming, vol 19. Siam
22. Diehl M, Ferreau HJ, Haverbeke N (2000) Efficient numerical methods for nonlinear MPC and moving horizon estimation. In: Nonlinear model predictive control, pp 391–417. Springer
23. Bae S, Kim Y, Guanetti J, Borrelli F, Moura S (2018) Design and implementation of ecological adaptive cruise control for autonomous driving with communication to traffic lights (2018). arXiv:1810.12442
24. Shaobing X, Peng H (2018) Design and comparison of fuel-saving speed planning algorithms for automated vehicles. IEEE Access 6:9070–9080
25. Kaijiang Y, Yang J, Yamaguchi D (2015) Model predictive control for hybrid vehicle ecological driving using traffic signal and road slope information. Control Theory Technol 13(1):17–28

26. Guillemin F, grondin O, Moulin P, Dib W (2015) An innovative eco-driving coaching solution: Geco. In: Proceedings of ITS world congress
27. Toffetti A, Iviglia A, Arduino C, Soldati M (2014) "ecodriver" HMI feedback solutions. In: Proceedings of the human factors and ergonomics society Europe chapter 2013 annual conference
28. Seewald P, Kroon L, Brouwer S et al (2015) Ecodriver. d13. 2: results of HMI and feedback solutions evaluations. Technical report, European Union
29. Hibberd DL, Jamson AH, Jamson SL (2015) The design of an in-vehicle assistance system to support eco-driving. Transp Res Part C: Emerg Technol 58:732–748
30. Vaezipour A, Rakotonirainy A, Haworth N (2016) Design of a gamified interface to improve fuel efficiency and safe driving. In: Proceedings of international conference of design, user experience, and usability, pp 322–332. Springer
31. IFP Energies Nouvelles (2018) Premier challenge national écoconduite interentreprises 2018: essai transformé! https://www.ifpenergiesnouvelles.com/node/616
32. Birrell SA, Fowkes M (2014) Glance behaviours when using an in-vehicle smart driving aid: a real-world, on-road driving study. Transp Res Part F: Traffic Psychol Behav 22:113–125
33. Birrell SA, Fowkes M, Jennings PA (2014) Effect of using an in-vehicle smart driving aid on real-world driver performance. IEEE Trans Intell Transp Syst 15(4):1801–1810
34. Borrelli F, Morari M (2007) Offset free model predictive control. In: Proceedings of conference on decision and control, pp 1245–1250. IEEE
35. Thrun S, Montemerlo M, Dahlkamp H, Stavens D, Aron A, Diebel J, Fong P, Gale J, Halpenny M, Hoffmann G et al (2006) Stanley: the robot that won the darpa grand challenge. J Field Robot 23(9):661–692