Improved Drive Cycle Following with an ILC Supported Driver Model

Lars Eriksson * **Mikael Norrlöf** **

* *Lars Eriksson is with Vehicular Systems, Dept. of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden,*
lars.eriksson@liu.se
** *Mikael Norrlöf is with Automatic Control, Dept. of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, and ABB AB – Robotics, SE-721 68 Västerås, Sweden,*
mikael.norrlof@liu.se

Abstract: Drive cycle following is important for concept comparisons when evaluating vehicle concepts, but it can be time consuming to develop good driver models that can achieve accurate following of a specific velocity profile. Here, a new approach is proposed where a simple driver model based on a PID controller is extended with an Iterative Learning Control (ILC) algorithm. Simulation results using a nonlinear vehicle and control system model show that it is possible to achieve very good cycle following in a few iterations with little tuning effort. It is also possible to utilize the repetitive behavior in the drive cycle to accelerate the convergence of the ILC algorithm even further.

Keywords: Vehicle control, learning, powertrain.

1. INTRODUCTION

Driving cycle following is a fundamental part in the evaluation of new powertrain solutions and their impact on for example fuel consumption and emissions. There are several cycles for example the European NEDC, the American FTP-75, and the comming WLTP. For a fair comparison between different concepts within each region it is fundamental that they follow the same driving cycle. In vehicle tests this can be done by a driver, a driver robot or in simulation by a driver model. Driver modeling is in itself a large subject, see Levermore et al. (2014) for a recent survey, where one application is speed following of a given trajectory. Differences in speed following can give rise to variations in fuel consumption and emissions, where an accurate and repeatable following can reduce the variations (Holschuh et al., 1991).

The focus here is on control of a vehicle so that it follows a prescribed velocity trajectory. The main contribution in this work is the implementation and demonstration of how a simple driver model that is augmented with the Iterative Learning Control (ILC) framework (Arimoto et al., 1984; Moore, 1993; A. Bristow et al., 2006) can quickly achieve good cycle following. This is the first application of ILC to vehicle speed tracking for drive cycle following.

2. BACKGROUND ON DRIVE CYCLE FOLLOWING

Drive cycle following is essentially an inverse problem where gas and brake pedal inputs are to be found (together with clutch and gear ratios) so that the vehicle follows the desired trajectory.

In the literature there are essentially two approaches, forward simulation and inverse systems modeling, sometimes called Quasi Steady Simulation (QSS). See Figure 1 for a schematic illustration of the two approaches, and Guzzella and Sciarretta (2007); Hofman et al. (2011) for more details.

In the *inverse system* or *QSS* approach, the system is modeled with the drive cycle as input, using a quasi steady state assumption for the acceleration between samples the required wheel force and powertrain torques are propagated through inverse models so that the desired velocity profile is followed. See Guzzella and Amstutz (1999); Wipke et al. (1999) for discussions and more details about QSS approach. Modeling with the inverse system often requires new tailored models to be developed, which might be non-trivial for complex systems and furthermore if there are non-minimum phase behaviors, the inverse is not stable and even more advanced techniques have to be employed, see Fröberg and Nielsen (2008) for a discussion.

The other technique, *forward simulation*, is to use a driver that tries to follow the specified trajectory, this is es-

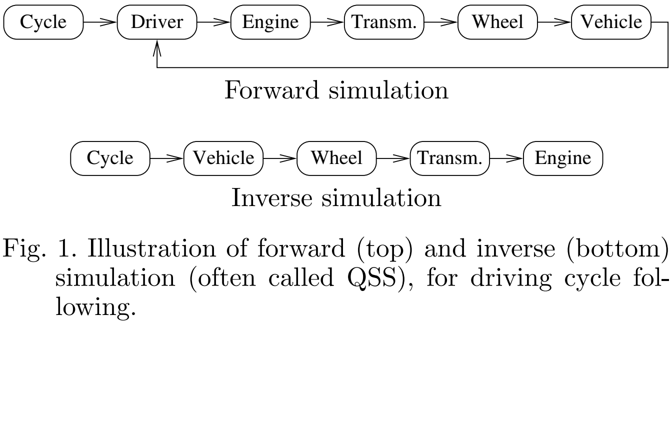

Fig. 1. Illustration of forward (top) and inverse (bottom) simulation (often called QSS), for driving cycle following.

[1] *This work was supported by the VINNOVA Industry Excellence Center LINK-SIC.

sentially a feedback control strategy where the speed is tracked using a controller that adjusts the pedals in response to tracking errors. An advantage with this approach is that no tailored model is necessary and it can cope with arbitrary model complexities. This can be applied both in simulation as well as to vehicle dynamometer tests. On the other hand, the quality, and the tuning, of the driver model will have an impact on how well the driving cycle can be followed. Concerning drive cycle following systems with forward approaches there are several papers that cover the mechatronic design of pedal robots and the control design for the servos that presses the pedals (Namik et al., 2006; Alt and Svaricek, 2010), while only a few discuss control for vehicle speed following (Chen et al., 2013; Eriksson, 2001).

In this paper the forward simulation technique is used, because it is versatile and easy to apply for different powertrain configurations. A further motivation for the ILC modification of the controller is that previous experience has shown that it is difficult to achieve good drive cycle following for general powertrains. To illustrate this point, the experience acquired during an engine concept evaluation done in Eriksson et al. (2012) is used. To get a good generic driver model that could cope with varying powertrain configurations required a development effort of about 6 months. It is desirable to cut this development time and have a generic driver model that can easily adapt to different drivetrains.

3. VEHICLE MODEL

Vehicle drivetrains are non-linear and complex, while the proposed PID & ILC based driver procedure is simple and partly relies on linear filters. Therefore, to test the driver model a complete vehicle model for longitudinal motion that includes significant nonlinear elements is implemented and described here. An illustration of the complete vehicle model is shown in Figure 2. The vehicle model is composed by:

- a naturally aspirated engine (rate limited throttle movement, filling and emptying intake manifold dynamics, volumetric efficiency for engine flow, torque model with ignition influence). See Figure 3 for the Simulink model of the engine.
- Driveline (clutch, stiff driveline, rolling conditions at the wheel). The clutch model contains the logics for break up and lock-up of the clutch and its logic keeps track of the wheel speed as well as the engine speed.
- Vehicle model (longitudinal mass model, rolling resistance, air drag model).

The vehicle model is standard and the reason for including the model description at this level of detail in this paper is to highlight that it includes significant nonlinearities as well as the time constants and time delays that can limit the gains that can be used in a feedback controller. In summary the model will not be used in a constructive way in the control design instead it will be used as a challenging test case for the approach and it is provided for completeness of the evaluation.

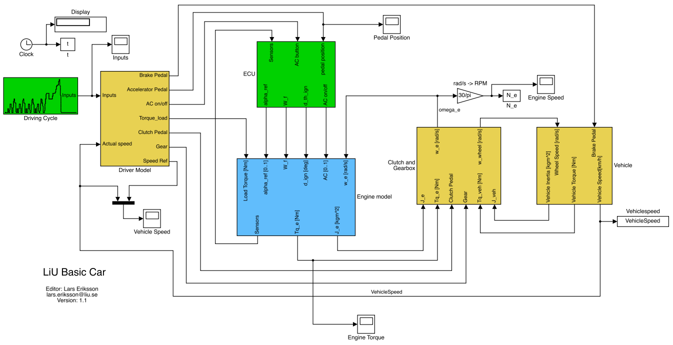

Fig. 2. Vehicle model implemented in Simulink. Containing drive cycle specification, driver model, Engine Control Unit (ECU), Engine model, Clutch and stiff driveline, and finally a Vehicle body model.

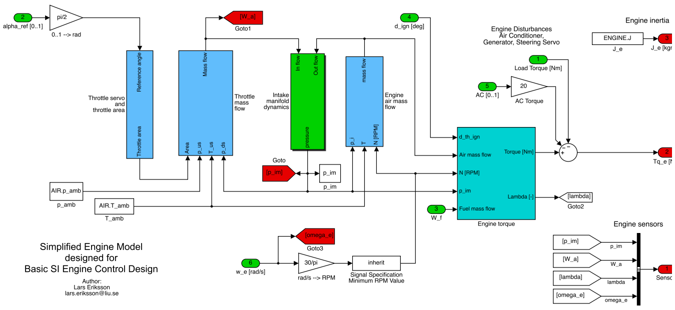

Fig. 3. Engine model inside the vehicle model implemented in Simulink. The engine model consists of a rate limiter for the throttle movement, a standard throttle flow model, intake manifold with isothermal filling and emptying, volumetric engine flow model and a torque model.

3.1 Engine Model

The control inputs to the engine model are throttle position reference α_{ref}, fuel mass flow $\dot{m}_{\mathrm{f}}$, ignition timing $\Delta\theta_{\mathrm{ign}}$, and additional loads such as air conditioner compressor on or off, u_{AC}, and auxiliary load torque M_{aux}. Another input to the engine model is the engine speed ω_{e}, provided as an exogenous input by the clutch and driveline model, this is motivated by the fact that the engine is not a completely autonomous system. In particular when the clutch is locked then the engine speed is dictated by the interaction between the engine model and the driveline and vehicle sub-models therefore the engine speed sub-model is placed in the clutch where the logics for lockup and break apart is implemented. The first input to the engine model, see Figure 3, is the throttle reference α_{ref}. This input goes through the throttle servo model which is modeled as a rate limiter and a conversion from $[0 \ldots 1]$ to radians. The throttle angle is then passed to a throttle area function, that gives the effective area

$$A(\alpha) = A_0 + A_{\max}\,(1 - \cos\,\alpha)$$

that has a leakage area A_0 and the maximum area $A_{\max} + A_0$ and a simple $\cos(\cdot)$ function for the angle to area dependence. The throttle mass flow is modeled with the standard isentropic restriction model,

$$\Pi(\frac{p_{\mathrm{ds}}}{p_{\mathrm{us}}}) = \max\left(\frac{p_{\mathrm{ds}}}{p_{\mathrm{us}}}, \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}}\right) \tag{1a}$$

$$\Psi_0(\Pi) = \sqrt{\frac{2\gamma}{\gamma-1}\left(\Pi^{\frac{2}{\gamma}} - \Pi^{\frac{\gamma+1}{\gamma}}\right)} \tag{1b}$$

$$\dot{m}_{\mathrm{at}}(p_{\mathrm{us}}, T_{\mathrm{us}}, p_{\mathrm{ds}}, A) = A\, C_D\, \frac{p_{\mathrm{us}}}{\sqrt{RT_{\mathrm{us}}}}\, \Psi_0(\Pi(\frac{p_{\mathrm{ds}}}{p_{\mathrm{us}}})) \tag{1c}$$

The air mass flow to the cylinders is modeled using the volumetric efficiency,

$$\dot{m}_{\mathrm{ac}}(N, p_{\mathrm{im}}, T_{\mathrm{im}}) = \eta_{\mathrm{vol}}(N, p_{\mathrm{im}}, p_{\mathrm{em}})\frac{V_{\mathrm{D}}\, N\, p_{\mathrm{im}}}{n_{\mathrm{r}}\, R\, T_{\mathrm{im}}} \tag{2a}$$

$$\eta_{\mathrm{vol}}(N, p_{\mathrm{im}}, p_{\mathrm{em}}) = C_{\eta_{vol}}(N)\frac{r_{\mathrm{c}} - \left(\frac{p_{\mathrm{em}}}{p_{\mathrm{im}}}\right)^{1/\gamma}}{r_{\mathrm{c}} - 1} \tag{2b}$$

where the sub-model for η_{vol} takes the load dependence into account. Here $C_{\eta_{vol}}(N)$ is modeled as a constant.

The intake manifold pressure dynamics is modeled using the well known isothemal model with filling and emptying of mass as the main dynamics,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = \frac{R\,T}{V}(\dot{m}_{\mathrm{at}} - \dot{m}_{\mathrm{ac}}) \tag{3}$$

which gives the pressure for the mass flow and torque models.

The engine torque is modeled using the three component model,

$$M_{\mathrm{e}} = \frac{W_{\mathrm{e}}}{n_{\mathrm{r}}\, 2\,\pi} = \frac{W_{\mathrm{i,g}} - W_{\mathrm{i,p}} - W_{\mathrm{fr}}}{n_{\mathrm{r}}\, 2\,\pi} \tag{4a}$$

$$W_{\mathrm{i,g}} = m_{\mathrm{f}}\, q_{\mathrm{LHV}}\, \tilde{\eta}_{\mathrm{ig}} \tag{4b}$$

$$W_{\mathrm{i,p}} = V_{\mathrm{D}}\, \mathrm{PMEP} \tag{4c}$$

$$W_{\mathrm{fr}} = 2\,\pi\, n_{\mathrm{r}}\, M_{\mathrm{fr}} = V_{\mathrm{D}}\, \mathrm{FMEP} \tag{4d}$$

where the following sub-models are used,

$$\tilde{\eta}_{\mathrm{ig}} = (1 - \frac{1}{r_{\mathrm{c}}^{\gamma-1}}) \cdot \min(1, \lambda_{\mathrm{c}}) \cdot \eta_{ign} \cdot \eta_{ig,ch} \tag{5}$$

$$\eta_{ign} = 1 - C_{ig,2} \cdot (\Delta\theta_{\mathrm{ign}})^2 \tag{6}$$

$$\mathrm{PMEP} = V_{\mathrm{D}}\,(p_{\mathrm{em}} - p_{\mathrm{im}}) \tag{7}$$

$$\mathrm{FMEP} = C_{\mathrm{fr,0}} + C_{\mathrm{fr,1}}\,\frac{60\,N}{1000} + C_{\mathrm{fr,2}}\left(\frac{60\,N}{1000}\right)^2 \tag{8}$$

Here the ignition efficiency is modeled using the deviation in ignition from the optimal $\Delta\theta_{\mathrm{ign}}$ and the constant is $C_{ig,2} = 4.316$, from Eriksson and Nielsen (2014). The friction model has parameters $C_{\mathrm{fr,0}} = 0.97 \cdot 10^5$, $C_{\mathrm{fr,1}} = 0.15 \cdot 10^5$, and $C_{\mathrm{fr,2}} = 0.05 \cdot 10^5$ from Heywood (1988). Finally there are delays included in the torque development, there is a powerstroke delay from the air and fuel induction and a powerstroke delay from the ignition timing.

From the engine torque the air condition compressor and auxiliaries load torques are subtracted from the torque M_{e},

$$M_{\mathrm{e,out}} = M_{\mathrm{e}} - M_{\mathrm{AC}} - M_{\mathrm{aux}} \tag{9}$$

where the AC torque is $M_{\mathrm{AC}} = 30$ [Nm]. The engine out torque is fed to the clutch and driveline model.

3.2 Vehicle Model

The vehicle velocity and driving distance is modeled using Newton's second law, where the tractive force from the wheel driveline is the driving input and the losses rolling resistance, and air drag oppose the movement.

$$m\frac{dv}{dt} = F_{\mathrm{t}} - F_{\mathrm{roll}} - F_{\mathrm{air}} \tag{10a}$$

$$\frac{ds}{dt} = v \tag{10b}$$

where m is the vehicle mass. F_{roll} and F_{air} are modeled as,

$$F_{\mathrm{roll}} = m\, g\,(f_{r,1} + f_{r,1}\, v) \tag{11}$$

$$F_{\mathrm{air}} = \frac{1}{2}\rho_{\mathrm{air}}\, A\, C_D\, v^2 \tag{12}$$

where a flat road is assumed. The tractive force comes from the wheel torque which is the transmission torque minus the applied brake torque, $F_{\mathrm{t}} = \frac{M_{\mathrm{t}} - M_{\mathrm{brake}}}{r_{\mathrm{w}}}$, where r_{w} is the wheel radius. We are also assuming rolling conditions so that

$$\omega_{\mathrm{w}}\, r_{\mathrm{w}} = v \tag{13}$$

3.3 Clutch and Transmission Model

The transmission connects the clutch to the wheels and it is modeled as a mass-less and loss-free transmission, with gear ratio i_{t}.

$$\omega_{\mathrm{c}} = i_{\mathrm{t}}\, \omega_{\mathrm{w}} \tag{14a}$$

$$M_{\mathrm{c}}\, i_{\mathrm{t}} = M_{\mathrm{w}} \tag{14b}$$

With the rolling condition (13) the vehicle dynamics model can be translated to an equivalent rotating system, see Eriksson and Nielsen (2014), and expressed as a rotating system at the clutch side of the transmission. The clutch connects the transmission side to the engine and the total system dynamics is then modeled as follows for the decoupled mode,

$$\frac{\mathrm{d}\omega_{\mathrm{e}}}{\mathrm{d}t} = \frac{1}{J_{\mathrm{e}}}\,(M_{\mathrm{e,out}} - M_{\mathrm{c}}) \tag{15a}$$

$$\frac{\mathrm{d}\omega_{\mathrm{c}}}{\mathrm{d}t} = \frac{i_t^2}{J_{\mathrm{w}} + m\, r_{\mathrm{w}}^2}\left(M_{\mathrm{c}} - \frac{1}{i_{\mathrm{t}}}(M_{\mathrm{brake}} - r_{\mathrm{w}}\,(F_{\mathrm{roll}} + F_{\mathrm{air}})\right) \tag{15b}$$

where M_{c} is the clutch torque and J_{e} and J_{w} are engine and wheel inertias. The clutch torque with lock-up and break apart is modeled according to Eriksson (2001). In clutch lockup mode the transferred clutch torque is calculated from considering a rigid body with total acceleration determined from the torque balance and total inertia giving the following expression for the clutch torque in lockup

$$M_{\mathrm{c}} = \frac{M_{\mathrm{e,out}}\, J_{\mathrm{v}} + M_{\mathrm{v}}\, J_{\mathrm{e}}}{J_{\mathrm{e}} + J_{\mathrm{v}}} \tag{16}$$

with the following intermediate variables

$$M_{\mathrm{v}} = \frac{1}{i_{\mathrm{t}}}(M_{\mathrm{brake}} - r_{\mathrm{w}}\,(F_{\mathrm{roll}} + F_{\mathrm{air}}) \tag{17}$$

$$J_{\mathrm{v}} = \frac{J_{\mathrm{w}} + m\, r_{\mathrm{w}}^2}{i_t^2} \tag{18}$$

Equation (16) gives accelerations of the engine and transmission sides that match and the engine and gearbox side thereby gets a loading/driving torque that is consistent with the acceleration. During gear changes, when the clutch decouples engine from the vehicle there is no connection to the wheels and the controller will not have any possibility to reduce the tracking error of the drive cycle. This is one of the main reasons for bringing up this submodel at this level of detail in this paper.

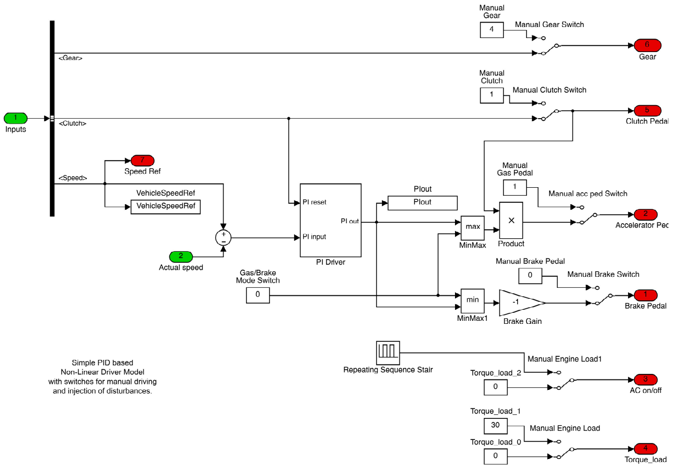

Fig. 4. Driver model inside the vehicle model implemented in Simulink.

3.4 ECU Model

In the ECU the control functions for fuel injection, throttle and ignition timing, including an idle speed controller are included. A torque based structure according to Eriksson and Nielsen (2014) is used, where there is a driver interpretation and translation to set-points for intake manifold pressure and finally throttle and ignition control inputs. The lambda controller is estimating the air flow to the cylinder via the volumetric efficiency and gives the fuel flow to the engine. Here it is assumed that the air fuel ratio controller works perfectly so there is no need for a feedback controller for λ.

4. SIMPLE PID DRIVER

The driver model, Figure 4, is the same as in Eriksson (2001). It is a simple but non-linear PID with speed reference, controlling the gas pedal and brake. In the controller the brake and gas pedal are not actuated at the same time and when the clutch is pressed the gas pedal is released. There is also a reset function of the integrator coupled to the clutch actuation which mimics that a driver starts over with the gas after changing the gear. Figure 5 shows the controller and the speed following when the simple PI driver follows the NEDC profile, note that no fine tuning has been performed since the purpose is not to make a benchmark between PID and ILC control only to apply ILC concept to drive cycle following.

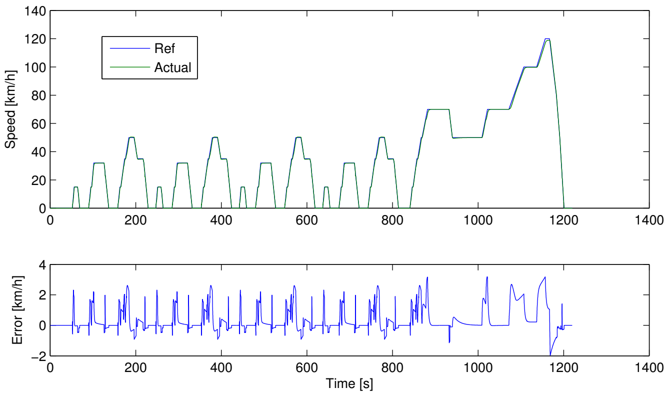

Fig. 5. Simulation results: Top– NEDC speed profile, Bottom – speed tracking error.

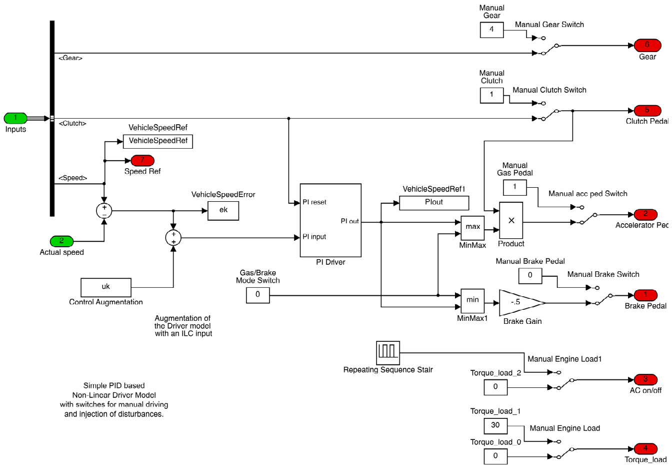

Fig. 6. Driver model augmented with ILC elements u_k and e_k, that is a sub-model of the complete vehicle model implemented in Simulink.

5. ILC IMPROVED DRIVER

The basic idea in the ILC framework (A. Bristow et al., 2006; Arimoto et al., 1984; Moore, 1993) is to use the error from one iteration of a repetitive action and based upon this information compute an update to the reference or control input to the system. Given an abstract system,

$$\begin{aligned}\dot{x}_k(t) &= f(x_k(t), u_k(t), r(t))\\ y_k(t) &= h(x_k(t))\end{aligned} \tag{19}$$

with x_k the system state, y_k the observed output, $r(t)$ the desired reference trajectory, t is the time and k the iteration number. In this paper the system represents the complete vehicle including the driver controller, see Sections 3 and 4. The input is therefore the speed reference and the output is the actual speed of the vehicle. In each iteration k, $t \in [0, T_f]$, for some final time T_f, and it is also assumed that the states are reset so that $x_k(0) = x_0(0)$ in all iterations $k > 0$. Here, what characterizes an iteration is that the reference is the same and since the initial state is also considered to be the same it is possible to iteratively update the input to achieve improved tracking. A general first order ILC algorithm is used (Moore, 1993; Norrlöf and Gunnarsson, 2002),

$$u_{k+1}(t) = Q(q)(u_k(t) + L(q)e_k(t)) \tag{20}$$

where q is the time shift operator, defined as $q^\kappa e(t) = e(t+\kappa T_s)$, T_s is the sample time for the discrete time ILC controller. The error is defined as $e_k(t) = r(t) - y_k(t)$, *i.e.*, the speed tracking error. The control signal $u_k(t)$ is applied as a correction term to the speed profile, which acts as the input to the driver model, see Figure 6 where the ILC input and the error computation are included explicitly in the model. In summary the error from a the most recent run k is filtered with $L(q)$ and added to the control signal and then both are smoothed with $Q(q)$ and used as correction term to the next run $k + 1$.

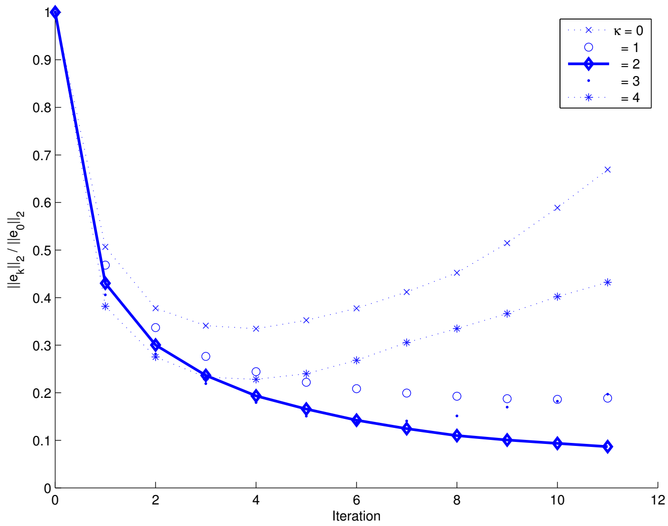

Fig. 7. Normalized 2-norm of the error, $\frac{||e_k||_2}{||e_0||_2}$, as a function of iteration for different choices of κ in the ILC algorithm. The solid line indicates the chosen κ value.

6. SIMULATION RESULTS

6.1 ILC algorithm tuning

In the simulations the sampling time is $T_s = 0.1$ s, the filter $Q(q)$ in (20) is chosen as a non-causal filter, realized as a discrete time second order Butterworth filter with cut-off frequency 2.5 Hz, applied using the filtfilt operator in MATLAB. $L(q)$ in (20) is chosen as $L(q) = \gamma q^{\kappa}$ and $u_0(t) = 0$, *i.e.*, no ILC compensation is applied in the initial iteration. To find the values of the parameters γ and κ some simple design rules are applied. The static gain from speed reference to actual speed is close to one, therefore the gain from error to speed reference compensation, γ, is chosen close to one, here $\gamma = 0.95$. Reducing γ will make the algorithm more robust to gain errors in the model but it will also give a slower convergence. A too high gain can result in overshoot in the compensation, possibly also to a divergent behavior. The parameter κ relates to the delay of the system and it is chosen to compensate for it in a non-causal way. With ILC it is possible to apply the error before it can be seen speed output and hence effectively compensate for the error in the system. In Figure 7 the 2-norm of the error is shown for different choices of κ in the interval $0, 1, \ldots, 4$ and the corresponding ∞-norm of the error is provided in Figure 8. With $\kappa = 2$ the 2-norm of the error is monotonically decreasing as a function of iteration which makes it a good choice for the design. Also the maximum error is decreasing, although $\kappa = 3$ actually has a faster convergence. Considering the 2-norm and the ∞-norm results together, still indicates that $\kappa = 2$ is a more robust solution since the error does not start to increase after some iterations.

6.2 Drive cycle properties

A first simulation of the vehicle over the complete New European Driving Cycle (NEDC) is shown in Figure 5. The top diagram shows the speed profile and the lower diagram shows the speed error. The PID driver model is just tuned to give a very rudimentary following of the cycle and the speed errors are quite large, about 2.5 km/h. The NEDC driving cycle contains four repetitive segments consisting of three different start, acceleration, and stop scenarios. At the end of the cycle a longer segment with higher speed is included.

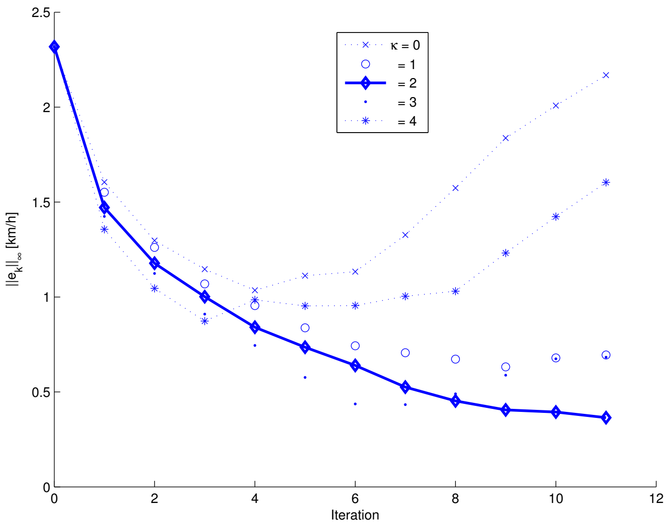

Fig. 8. Maximum error, $||e_k||_\infty$, as a function of iteration for different values of the parameter κ in the ILC algorithm. The solid line indicates the result when $\kappa = 2$.

With iterative learning control the repetitive actions can be improved using information from previous iterations of the same action. From Figure 5 it is clear that within one driving cycle it is possible to have four iterations of ILC, considering the first repetitive segments in the cycle. The first time the cycle is executed, considering the part of the cycle shown in Figure 9, iterations 0, 1, 2, and 3 are therefore performed. The second time, iterations 4 to 7 are performed, and so on. In the simulations the result from running three drive cycles, *i.e.*, iterations 0 to 11 in the ILC algorithm, are shown.

6.3 NEDC Results using ILC

When applying the ILC algorithm the speed profile converges to the desired profile in a few iterations. Figure 9 shows the vehicle desired and actual speed for the first section of the cycle and the speed profiles are seen to quickly converge to the desired speed.

The final level of the 2-norm after 11 iterations is less than 10% of the value without ILC. Figure 10 shows the time profile of the speed error from Figure 9 without ILC, after 3 ILC iterations or one full drive cycle, after 7 ILC iterations or two full drive cycles, and after 11 iterations or three full drive cycles. In Figure 8 the maximum error is shown and the error is decreased from a level of nearly 2.5 km/h to below 1 km/h after 3 ILC iterations, *i.e.*, one full drive cycle. This is achieved with very little development and tuning effort put on the driver model and also on the ILC algorithm parameters. Analyzing the residual one sees that the maximum error are attained around the gear changes where the clutch is engaged and there is no way of influencing the vehicle speed with the gas pedal as it is not an available torque input in the decoupled mode.

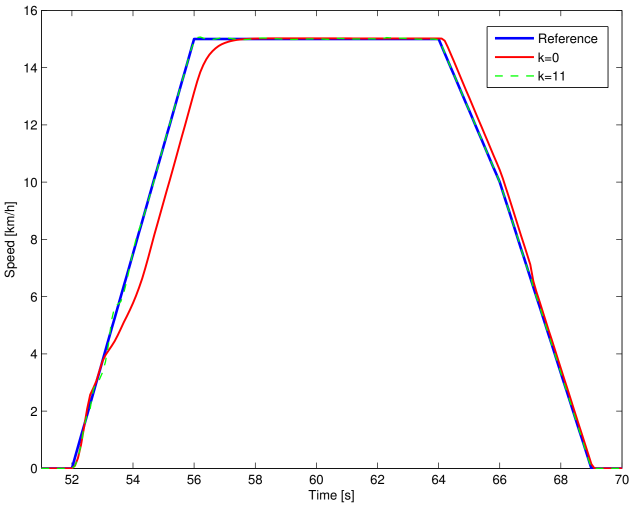

Fig. 9. Speed tracking for the ILC iterations, showing how the speed converges to the desired speed profile, reference speed (solid blue), nominal speed profile, y_0 (solid red), and finally, the speed profile after 11 iterations, y_{11} (dashed green). The ILC algorithm uses $\kappa = 2$.

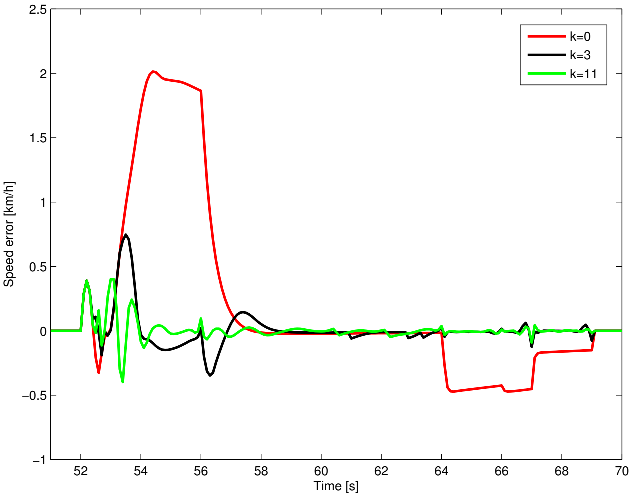

Fig. 10. Speed tracking for the ILC iterations, showing how errors decrease and come well below the desired 2 $^{\mathrm{km}}/_{\mathrm{h}}$ error, nominal error without ILC, e_0 (red), after 3 iterations, e_3 (blue), and after 11 iterations, e_{11} (green). The ILC algorithm uses $\kappa = 2$.

6.4 FTP-75 Results using ILC

The NEDC cycle used in the results in Figures 9 to 8 is regular in shape and repetitive and it is therefore of interest to apply it to other less smooth and repetitive cycles. Therefore results from applying ILC with the same tuning as above to the FTP-75 cycle are shown in Figures 11 to 13. The full cycle is shown in Figure 11 and an enlargement of the first section is shown in Figure 12. Figure 13 shows that the initial error is 6.2 $^{\mathrm{km}}/_{\mathrm{h}}$ (but it must be noted that the PID has not been tuned and optimized) as the focus in the paper is to study how the ILC behaves. The ILC converges fairly quickly and comes within 1 $^{\mathrm{km}}/_{\mathrm{h}}$.

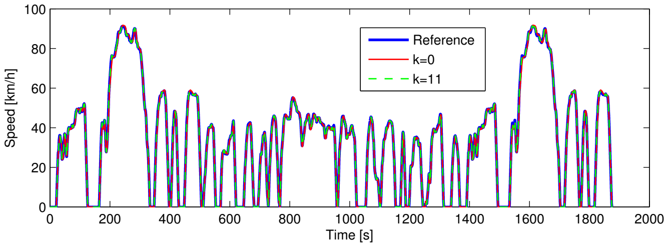

Fig. 11. Speed profile for the FTP-75 cycle with the initial and final tracking.

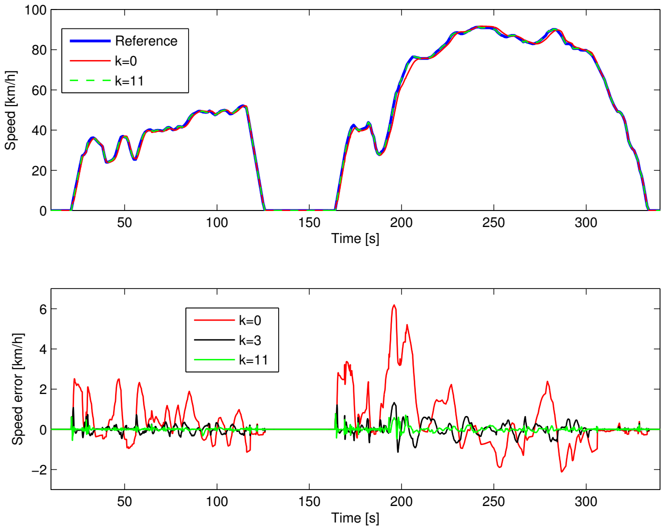

Fig. 12. Speed tracking and errors for the first 340 s section of the FTP-75. The main remaining errors occur around gearshifts.

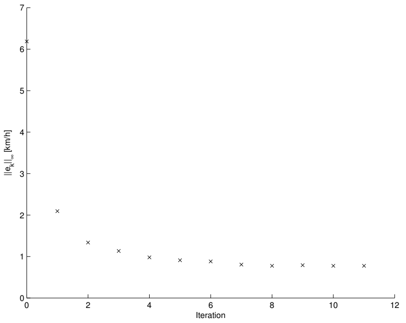

Fig. 13. Maximum error as function of iteration number. It starts with 6.2 $^{\mathrm{km}}/_{\mathrm{h}}$ and is decreased down to under 1 $^{\mathrm{km}}/_{\mathrm{h}}$ after 11 iterations.

Analyzing the residual after the learning one sees also for the FTP-75 that the maximum errors are attained around the gear changes where the clutch is engaged. In fact continuing the learning beyond 11, with the current tuning, increases the error as the errors around gearchanges are spread out due to the smooting filter.

Even though the FTP-75 is less regular than the NEDC there is a possibility to introduce some learning during the cycle. In Figure 11 it is seen that the intial 500 s are repeated at the end of the cycle where learning can be introduced and repeated.

6.5 Future work

In the future, extensions to cope with a decoupled driveline are planned. As pointed out in Figure 7 and Figure 8 the speed errors are decreasing with the iterations. There are, however, some occasions where the error cannot decrease, due to the physics of the driveline. In particular, when the driveline is decoupled during a gear change, there is no possibility to affect the vehicle speed during a short period of time. The ILC algorithm used in the simulation example is a standard first order ILC algorithm. Since the system dynamics is time and state dependent it would be natural to use an optimization based ILC algorithm (Gunnarsson and Norrlöf, 2001) where time dependent weights can be used in order to reduce the ILC gain in conditions where the engine cannot affect the vehicle speed.

In addition, next step involves to implement algorithms in an experimental platform and perform experiments in both the vehicular systems engine lab as well as the vehicular systems vehicle propulsion lab that has a complete vehicle driveline with a driver robot, see Öberg et al. (2013) for a description of the environment. The result is simple to implement as the output from the learning algorithm $u_k(t)$ is added directly to the desired speed profile that is sent as a command to the driver robot that controls the gas and brake pedals.

7. CONCLUSIONS

A successful demonstration of the improvement that ILC can give for vehicle speed following in driving cycle simulations has been shown. The vehicle model used for longitudinal motion is non-linear and contains the most significant dynamics and limitations. The ILC algorithm used is of standard first order type, with a low pass filtering and the result is achieved with a minimum effort in tuning of the parameters in both the driver controller and the ILC algorithm. Faster convergence can be achieved using more advanced ILC design schemes but the desired maximum $2\,{}^{\mathrm{km}}/_{\mathrm{h}}$ speed error with the proposed approach is achieved already after one iteration on the NEDC. The repetitive pattern in the NEDC drive cycle was utilized to increase the convergence speed of the ILC algorithm on a part of the cycle. The same drive pattern is repeated four times in the cycle and by utilizing this in the algorithm multiple ILC iterations can be performed during the same drive cycle.

REFERENCES

A. Bristow, D., Tharayil, M., and G. Alleyne, A. (2006). A survey of iterative learning control - a learning-based method for high-performance tracking control. *IEEE Control Systems Magazine*, 96–114.

Alt, B. and Svaricek, F. (2010). Second-order sliding modes control for in-vehicle pedal robots. In *Variable Structure Systems (VSS), 2010 11th International Workshop on*, 516–521. doi:10.1109/VSS.2010.5544545.

Arimoto, S., Kawamura, S., and Miyazaki, F. (1984). Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2), 123–140.

Chen, G., Zhang, W.g., and Zhang, X.n. (2013). Speed tracking control of a vehicle robot driver system using multiple sliding surface control schemes. *International Journal of Advanced Robotic Systems*, 10.

Eriksson, L. (2001). Simulation of a vehicle in longitudinal motion with clutch engagement and release. In *3rd IFAC Workshop "Advances in Automotive Control" Preprints*. Karlsruhe, Germany.

Eriksson, L., Lindell, T., Leufven, O., and Thomasson, A. (2012). Scalable component-based modeling for optimizing engines with supercharging, E-boost and turbocompound concepts. *SAE International Journal of Engines, Paper 2012-01-0713*, 5(2), 579–595.

Eriksson, L. and Nielsen, L. (2014). *Modeling and Control of Engines and Drivelines*. John Wiley & Sons.

Fröberg, A. and Nielsen, L. (2008). Efficient drive cycle simulation. *IEEE Transactions on Vehicular Technology*, 57(2), 1442–1453.

Gunnarsson, S. and Norrlöf, M. (2001). On the design of ILC algorithms using optimization. *Automatica*, 37(12), 2011–2016.

Guzzella, L. and Amstutz, A. (1999). Cae tools for quasi-static modeling and optimization of hybrid powertrains. *IEEE Transactions on Vehicular Technology*, 48(6), 1762–1769.

Guzzella, L. and Sciarretta, A. (2007). *Vehicle Propulsion Systems – Introduction to Modeling and Optimization*. Springer Verlag, 2 edition.

Heywood, J.B. (1988). *Internal Combustion Engine Fundamentals*. McGraw-Hill series in mechanical engineering. McGraw-Hill.

Hofman, T., Leeuwen, D.V., and Steinbuch, M. (2011). Analysis of modelling and simulation methodologies for vehicular propulsion systems. *International Journal of Powertrains*, 1(2), 117–136.

Holschuh, N., Winckler, J., Probst, H., and Glinski, K.v. (1991). Performance of a mechanized driver for measurements of automobile exhaust gas emissions and fuel economy on chassis dynamometer. Technical report, SAE Technical Paper 910037.

Levermore, T., Ordys, A., and Deng, J. (2014). A review of driver modelling. In *Control (CONTROL), 2014 UKACC International Conference on*, 296–300. doi: 10.1109/CONTROL.2014.6915156.

Moore, K.L. (1993). *Iterative Learning Control for Deterministic Systems*. Advances in Industrial Control. Springer-Verlag, London.

Namik, H., Inamura, T., and Stol, K. (2006). Development of a robotic driver for vehicle dynamometer testing. In *Proceedings of 2006 Australasian Conference on Robotics and Automation. Auckland, New Zealand*.

Norrlöf, M. and Gunnarsson, S. (2002). Experimental comparison of some classical iterative learning control algorithms. *IEEE Transactions on Robotics and Automation*, 18(4), 636–641.

Öberg, P., Nyberg, P., and Nielsen, L. (2013). A new chassis dynamometer laboratory for vehicle research. *SAE International Journal of Passenger Cars - Electronic and Electrical Systems*, 6(1), 152–161.

Wipke, K.B., Cuddy, M.R., and Burch, S.D. (1999). Advisor 2.1: A user-friendly advanced powertrain simulation using a combined backward/forward approach. *IEEE Transactions on Vehicular Technology*, 48(6), 1751–1761.