Linköping Studies in Science and Technology
Thesis No. 1181

Extending the Inverse Vehicle Propulsion Simulation Concept -To Improve Simulation Performance

Anders Fröberg

Linköpings universitet
INSTITUTE OF TECHNOLOGY

Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden

Linköping 2005

Extending the Inverse Vehicle Propulsion Simulation Concept
-To Improve Simulation Performance

froberg@isy.liu.se
http://www.fs.isy.liu.se/
Department of Electrical Engineering,
Linköpings universitet,
SE-581 83 Linköping,
Sweden.

ISBN 91-85299-83-9
ISSN 0280-7971
LiU-TEK-LIC-2005:36

To Carolina and Viktor

Abstract

Drive cycle simulations of longitudinal vehicle models is an important tool for design and analysis of power trains. On the market today there are several tools for such simulations, and these tools use mainly two different methods of simulation, forward dynamic or quasi-static inverse simulation. Forward dynamic simulation is capable of describing the dynamic behavior of a system to a high level of detail, but suffers from long simulation times. On the other hand, quasi-static inverse simulations are very fast, but lack the ability of describing additional dynamics in a good way. Here known theory for stable inversion of non linear systems is used in order to try to combine the fast simulation times of the quasi-static inverse simulation with the ability of describing the dynamics as in the forward dynamic simulation. The stable inversion technique together with a new implicit driver model forms a new concept, inverse dynamic simulation. Using this technique the need to develop dedicated inverse models is reduced, and it is shown that a large class of models that can be simulated in forward dynamic simulation also can be simulated in inverse dynamic simulation. In this respect, three powertrain applications are used that include important dynamics that can not be handled using quasi-static inverse simulation. The extensions are engine dynamics, drive line dynamics, and gas flow dynamics around diesel engines. These three cases also represent interesting mathematical properties such as zero dynamics, resonances, and non-minimum phase systems, i.e. unstable zero dynamics. The inversion technique is demonstrated on all three examples, and the feasibility of inverse dynamic simulation of these systems is shown. Moreover, using the three examples, inverse dynamic simulation is compared to forward dynamic simulation regarding simulation set-up effort, simulation time, and parameter-result dependency. It is shown that inverse dynamic simulation is easy to set up, gives short simulation times, and gives consistent result for design space exploration. This makes inverse dynamic simulation a suitable method to use for drive cycle simulation, and especially in situations requiring many simulations, such as optimization over design space, powertrain configuration optimization, or development of powertrain control strategies.

Acknowledgments

This work has been carried out under the guidance of Professor Lars Nielsen at Vehicular Systems, Department of Electrical Engineering, Linköpings Universitet, Sweden.

The Swedish Energy Agency, through the Center for Automotive Propulsion Simulation CAPSIM, and the Swedish Foundation for Strategic Research, through the Visimod project, are greatfully acknowledged for their funding.

I would like to express my gratitude to a number of people:

Professor Lars Nielsen for letting me join this group, for supervision and guidance of this work, and for the many interesting discussions along the way.

Erik Frisk, Jan Åslund, Johan Whalström and Per Öberg for proofreading the manuscript. Lars Nielsen, Erik Frisk, Jan Åslund, and Lars Eriksson for interesting research discussions. Per Andersson for the joyful discussions about all that is car related. The rest of the vehicular systems group for creating a nice atmosphere to work in.

Especially, I would like to thank my wife, Carolina, and our son, Viktor, for sharing life with me and bringing me joy and happiness. I love you!

Linköping, May 2005

Anders Fröberg

Contents

1

Introduction

Simulation of longitudinal vehicle models is a commonly used tool for driveline design, driveline optimization, and, design of driveline control strategies. There are two common ways to do this, quasi-static as, e.g., in Advisor and QSS-TB [33, 15], and forward dynamic simulation as, e.g. in PSAT and Capsim, [26, 35]. The quasi-static simulation uses vehicle speed and acceleration to calculate required torques and speeds backwards through the driveline. Finally fuel flow is calculated. Often the model consists of static equations and maps of the efficiency of the components. A major advantage of this method is that simulation time is so low that it can be used in design exploration and optimization loops. On the other hand, in forward dynamic simulation, differential equations are solved using, e.g., throttle position (or fuel flow) as input, and vehicle speed as output. These kind of models also requires a controller, a driver model, to track a given speed trajectory (drive cycle). The differential equations that then have to be solved typically gives an order of magnitude longer simulation times for drive cycle simulations than what is typical for Advisor or QSS. However, more effects can be included making the modeling more accurate than in quasi-static simulation.

It would of course be of considerable value to be able to extend the time-efficient quasi-static simulation with important dynamics without significantly loosing simulation performance. The goal of the thesis is to find a simulation method with such properties. Guiding principles in this development have been to make extensions to the quasi-static method that are suffi-

ciently general to include dynamics of the powertrain, such that important transients can be captured in the simulation. The method should also be fast enough to be part of optimization loops. The proposed simulation method to handle these demands is inverse dynamic simulation. The inverse dynamic simulation is quite similar to the quasi-static approach, but extends its use to also include dynamics in addition to the stationary maps. The vehicle speed, and derivatives thereof, are used as inputs to calculate required fuel flow.

1.1 Thesis organization and contributions

Since the requirements on the proposed method come from applications, the presentation is naturally structured by giving the background, proposing the new method and then studying it in simulations where also comparisons are made. First, in Chapter 2 a background of drive cycle simulation is given, where it is stated what demands users have on the simulation, and this leads to the problem formulation. The first step in the proposed method of inverse dynamic simulation is to invert the system to be simulated, and known theory for stable inversion of non linear systems is summarized in Chapter 3. To make a drive cycle simulation the vehicle has to track a given reference speed profile. In Chapter 4 driver models to obtain this tracking, for both forward and inverse dynamic simulation, are discussed. The contribution there is the implicit driver model used in inverse dynamic simulation.

It is not trivial to compare different simulation methods with each other. This is discussed in Chapter 5 where also, guided by the applications, some performance measures are defined that gives the basis for comparisons made. To compare the methods, and to demonstrate the feasibility of inverse dynamic simulation, three important powertrain applications are presented in Chapter 6. These examples represent three major transient dynamics that are important extensions to a basic driveline, and they also represent different mathematical characteristics like resonances, zero dynamics, and non minimum phase behavior. The powertrain examples are used in Chapter 7 to demonstrate feasibility, and to study simulation performance and usability of forward and inverse dynamic simulation. In Chapter 8 the conclusions of the thesis are drawn.

Part of the material presented in this thesis have been published in the papers:

A. Fröberg and L. Nielsen. *A Method to Extend Inverse Dynamic Simulation of Powertrains with Additional Dynamics.* First IFAC Symposium on Advances in Automotive Control. 2004.

A. Fröberg and L. Nielsen. *Dynamic Vehicle Simulation -Forward, Inverse and New Mixed Possibilities for Optimized Design and Control.* SAE Technical Paper Series SP-1826, 2004-01-1619. 2004.

2

Drive Cycle Simulation

Drive cycle simulations of longitudinal vehicle models is an important tool for driveline design, driveline optimization, and, design of driveline control strategies. A major objective in powertrain design is to minimize fuel consumption and component costs, while maximizing drivability. One reason to use drive cycle simulations in the design is emission legislations. The vehicle manufacturers have to certify their vehicles for emission outputs. This is typically done by driving the vehicle on a chassis dynamometer according to a specified speed profile, a drive cycle. During the test, emissions are collected in bags. In order to put a vehicle on the market, the emissions during a drive cycle test have to be below prescribed limits. Some emissions, like, e.g., CO_2, is closely coupled to fuel consumption. This test procedure, other demands to decrease fuel consumption, customers demands on driving comfort, costs, and so forth, put high demands on todays vehicles. Since it is very time demanding and cost ineffective to build prototypes in early stages of the development, drive cycle simulation has become a necessary tool in powertrain design and control system development.

This chapter gives the background of drive cycle simulation of today, naturally leading to the problem formulation of the thesis. Drive cycles are defined in Section 2.1. Tracking of a drive cycle in vehicle simulation is discussed in Section 2.1.1. The area of usage for drive cycle simulations is discussed in Section 2.2. Existing tools and methods for drive cycle simulation are discussed in Section 2.3. Advantages and disadvantages with todays sys-

tem are discussed in Section 2.4, and that naturally gives the goals and problem formulation of the thesis that is outlined in Section 2.5.

2.1 Drive cycles

A drive cycle is a speed profile where speed is defined as a function of time. Several cycles have been developed by governments around the world as a tool for vehicle certification. Examples of such cycles are the New European Drive Cycle, NEDC, EU Directive 2001/116/EC, 98/69/EC, and, EEC Directive 90/C81/01, see Figure 2.1, and the Federal Test Procedure FTP 75, see Figure 2.2, used by the US government. Other drive cycles are also used by the industry for testing of virtual prototypes.

2.1.1 Tracking requirements

As can be seen in Figures 2.1 and 2.2, a drive cycle can not be expected to be continuously differentiable. This means that it is impossible to follow, or track, a drive cycle exactly with finite torques and accelerations in the powertrain. Hence, in drive cycle simulation the speed has to be controlled to track the drive cycle in a desired way. When drive cycles are used for emission legislations the cycle has to be tracked by a human driver within certain limits. It is hence logical to track cycles within defined limits in simulation also. Tracking in simulation is achieved using a driver model that controls the vehicle models. The driver models has to be designed such that the

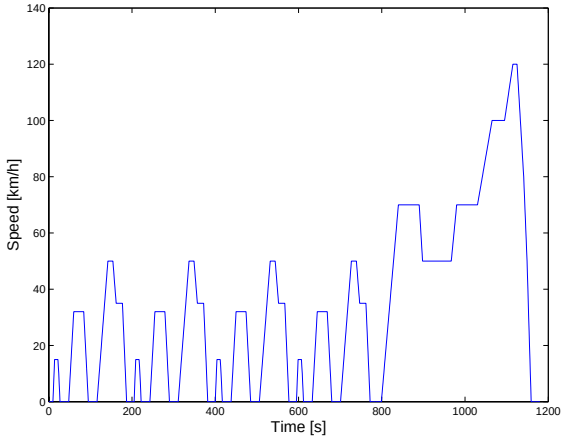

Figure 2.1 The New European Drive Cycle, NEDC, consisting of four ECE cycles and one EUDC cycle. Each ECE cycle is 195 s and 1.013 km long and is intended to resemble urban driving as it typically is in metropolitan areas in Europe such as Rome or Paris. The EUDC cycle is 400 s and 6.955 km long and is intended to resemble extra urban driving. EEC Directive 90/C81/01.

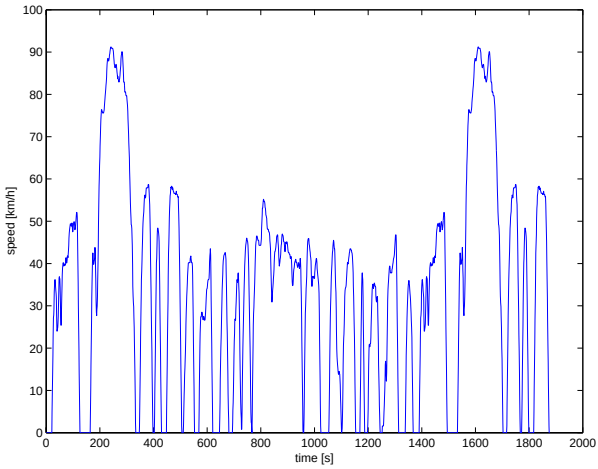

Figure 2.2 The Federal Test Procedure, FTP75, is a urban route of 1874 s and 17.77 km. The cycle is derived from FTP72 and consists of three phases, a cold start phase, 0-505 s, a transient phase, 505-1369 s, and a hot start phase, 1369-1874 s.

tracking is within the prescribed speed limits. In forward dynamic simulation this is usually done using a driver model consisting of a controller that controls the vehicle behavior. In inverse dynamic simulation tracking is achieved using drive cycle smoothing where the driver model specifies how the drive cycle is smoothed, see Section 4.2. As will be shown later, results in for example fuel consumption can differ a lot between two different velocity trajectories that both are within prescribed limits, see Figure 2.3. Hence, it is important to study how a drive cycle is tracked and not only that the tracking is within limits.

2.2 Usage of drive cycle simulation

There are a number of important users of drive cycle simulations, and among all applications there are many where simulation time is important. A lot of work has been done on developing drive cycle simulation tools such as Advisor and PSAT, see Figures 2.4 and 2.5, that have been used a lot in for example concept studies. Other examples of where drive cycle simulation are used is in optimization over design space [1], optimization of powertrain configuration [27], and design of powertrain control systems, [3, 13, 28]. Drive cycle simulation is also used in controllers such as model predictive cruise controllers. The type of inverse dynamic simulation described later can also be used as a feed forward part of non-linear powertrain con-

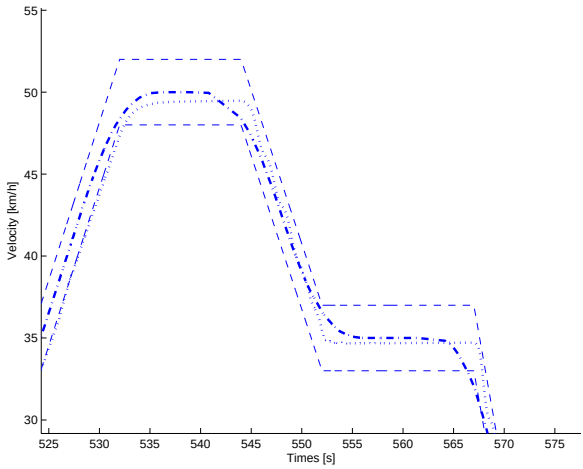

Figure 2.3 Two different simulations with different tracking of the NEDC cycle. Both simulations have tracking that is within specified limits, but they will result in different fuel consumption.

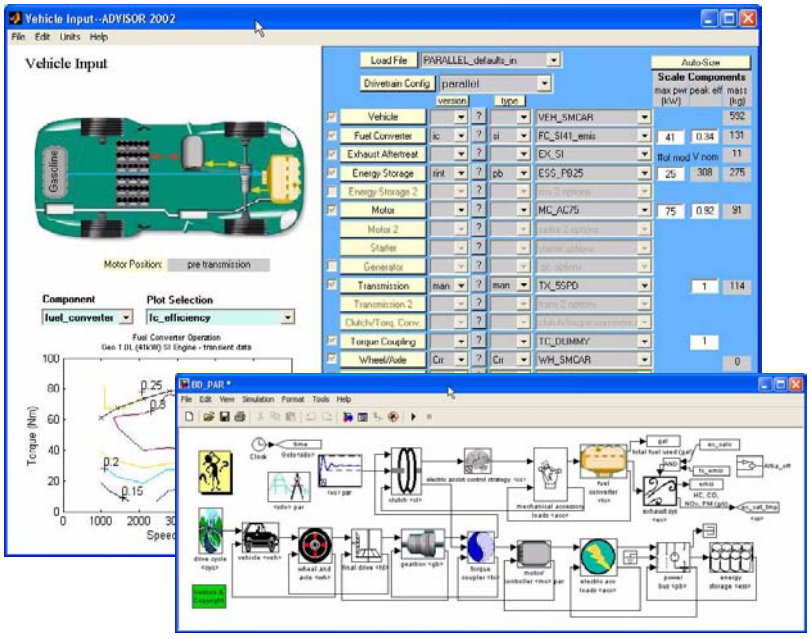

Figure 2.4 A screen dump from Advisor [33], a powertrain simulation tool that uses a quasi-static inverse method. The screen dump is from a model of a parallel hybrid electric vehicle.

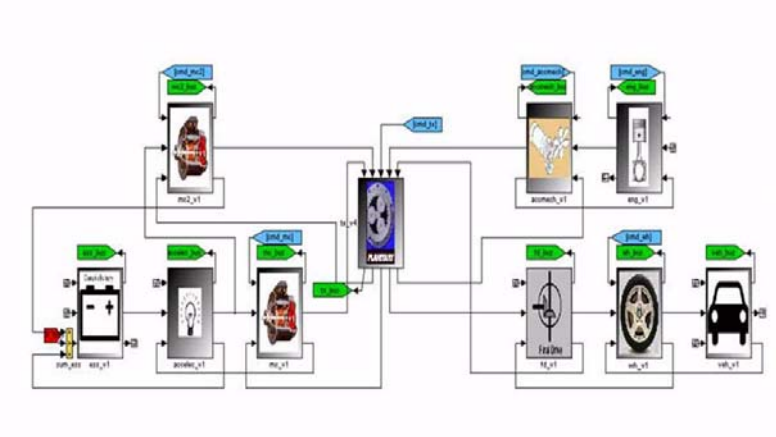

Figure 2.5 A screen dump from PSAT [26], a powertrain simulation tool that uses the forward dynamic simulation method. The screen dump is from a model of a power split hybrid electric vehicle.

trollers. In the applications mentioned above there are several additional requirements, besides simulation time, and this will be discussed more in Chapter 5.

2.3 Simulation techniques and tools

As mentioned before, there are several tools for simulation of longitudinal vehicle models on the market today. These tools use mainly two different methods of simulation, forward dynamic simulation and quasi-static inverse simulation. Forward dynamic simulation is capable of describing the dynamic behavior of a system to a high level of detail, but suffer from long simulation times. The quasi-static inverse simulations on the other hand are very fast, but lack the ability of describing the dynamics in a good way. Example of quasi-static inverse tools are, e.g., Advisor [33] and QSS-TB [15], and examples of forward dynamic simulation tools are, e.g., PSAT [26], Capsim [35], and, V-Elph [5]. All these tools use Matlab/Simulink. The coming subsections will explain the basis for these techniques and tools.

2.3.1 Basic modelling

As an example to illustrate the basic concepts, study the road load equation directly obtained from Newton's second law

$$m\dot{v}(t) = F(t) - mgc_r - \frac{1}{2}\rho c_d A v^2(t) \tag{2.1}$$

where the vehicle's mass times acceleration equals the propulsive force reduced by roll- and air resistance. This system will be used in the following subsections to exemplify the computational steps of the simulation methods.

2.3.2 Forward dynamic simulation

The most common way to model dynamic systems is as a system of first order ordinary differential equations, ODEs, such as

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ y(t) &= h(x(t), u(t)) \end{aligned} \tag{2.2}$$

where $x(t), u(t)$ and $y(t)$ are vectors. Forward dynamic simulation can be interpreted as: If a system being in a given state, x_0, is exposed to the external influence $u(t)$, how will the state $x(t)$ and the outputs $y(t)$ of the system be affected. Using this interpretation, forward dynamic simulation of systems like (2.2) is about solving initial value problems, IVPs. The numerical solution to the problem (2.2) is thoroughly described in the literature, see e.g. [2, 14, 16].

As an example of forward dynamic simulation, recall the system (2.1). The input to this system is the driving force $F(t)$ and the output is the vehicle speed $v(t)$. First the system is written in the form (2.2) for the derivative of the speed resulting in

$$\dot{v}(t) = \frac{1}{m}\left(F(t) - mgc_r - \frac{1}{2}\rho c_d A v^2(t)\right) \tag{2.3}$$

Then, given an initial value of the vehicle speed $v(t_0)$, and the input $F(t)$, the system (2.3) can be numerically integrated to compute the speed trajectory. In general, forward dynamic vehicle simulation is about solving differential equations, using, e.g., throttle position (or fuel flow) as input, and vehicle speed as output. These kind of models also requires a controller, a driver model, to track a given speed trajectory (drive cycle). See Figure 2.6 for a typical computational scheme of the forward dynamic simulation.

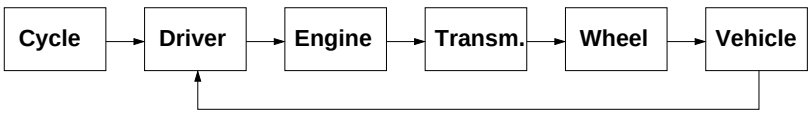

Figure 2.6 Schematic depiction of the computational scheme in forward dynamic simulation as in e.g. the Matlab/Simulink tool PSAT. Drive cycle tracking is implicit since it is obtained using an explicit driver model that controls the input to get the desired speed.

2.3.3 Inverse simulation

In inverse simulation a different approach is taken in that the input corresponding to a given output is calculated, i.e. $u(t) = f(x(t), \dot{x}(t))$.

For the system (2.1) that means that given speed v and acceleration $\dot{v}$, the driving force F is computed as

$$F(t) = m\dot{v}(t) + mgc_r + \frac{1}{2}\rho c_d A v^2(t) \tag{2.4}$$

Thus, given a drive cycle, velocity and acceleration are used to calculate required torques and speeds backwards through the driveline. This is done in order to calculate the required energy input to the system, making the vehicle follow the prescribed velocity profile.

Quasi-static approach. In existing tools today a quasi-static approach [15, 33] to inverse simulation is taken. In that approach to simulate the system (2.1), the speed $v(t)$ and acceleration are approximated as quasi-static by

$$v(t) = \frac{v(kh+h) + v(kh)}{2}, \forall t \in [kh, kh+h)$$
$$\dot{v}(t) = \frac{v(kh+h) - v(kh)}{h}, \forall t \in [kh, kh+h)$$

which are then inserted in the right hand side of (2.4). When solving (2.4) in this way only static equations are solved when the driving force F is computed. See Figure 2.7 for a typical computational scheme of quasi-static inverse simulation.

Figure 2.7 Schematic depiction of the computational scheme in quasi static simulation as in Advisor and the QSS-Toolbox. The simulated states are calculated back-wards from the drive cycle. No driver model is needed.

Extending quasi-static simulation. The example system (2.1) has only one state. If however more dynamics are to be added to the models, following the inverse simulation strategy, more states z have to be included. They have to be obtainable from the velocity profile, which means that higher derivatives of the speed, or drivecycle, may be needed. This can formally be written as $u(t) = f(x(t), \dot{x}(t), z(t)) = f(x(t), \dot{x}(t), \ddot{x}(t), \ldots)$. Such simulation, inverse dynamic simulation, is the main topic of this thesis as will be discussed in Chapters 3 and 4.

Since drivecycles can not be expected to be continuously differentiable, the procedure is not as straight forward. In order to add more dynamics some other method has to be used. Here, one such method, stable inversion of nonlinear systems, will be used. This method is described in [7] and [19] and related theory is presented in [20]. A summary of the technique is given in Chapter 3.

2.4 Characteristics

The quasi-static inverse simulation is a successful method to make fast simulations of powertrains. Often the model consists of static equations and maps of the efficiency of the components. A major advantage of this method is that simulation time is so low that it can be used in design exploration and optimization loops.

The differential equations that have to be solved in forward dynamic simulation typically give an order of magnitude longer simulation times for drive cycle simulations than what is typical for Advisor or QSS-TB. On the other hand, more effects can be included making the modeling more accurate than in quasi-static inverse simulation.

The characteristics of respective method are depicted in Figure 2.8. This figure also depicts the goal of the thesis, namely to combine the favorable characteristics of each method in a new method, inverse dynamic simulation.

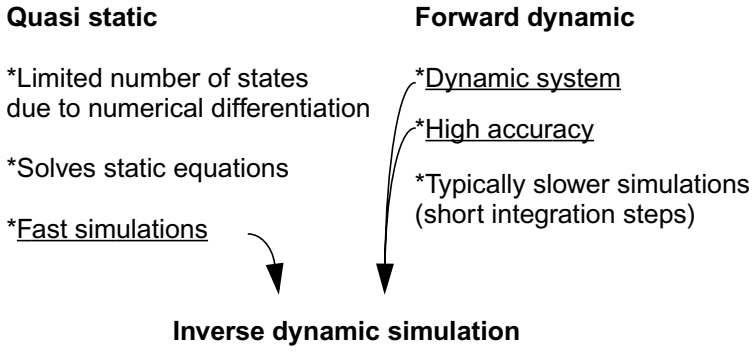

Figure 2.8 The characteristics of the quasi-static inverse simulation and the forward dynamic simulation. The ambition is to combine the advantages in a new method, inverse dynamic simulation.

2.5 Problem formulation

Having stated the main objective of the thesis, i.e. combining the good properties of quasi-static and forward dynamic simulation, in the previous section and in Figure 2.8, there are still a number of ramifications of the problem that has to be considered. The main ones are:

- When studying important properties of vehicles such as fuel consumption and emissions, engine and driveline dynamics has a big influence. Hence, it has to be shown that it is possible to extend the time-efficient quasi-static simulation with important dynamics without significantly losing simulation performance.

- The inversion of the non linear system, including the added dynamics, has to be performed.

- As depicted in Figure 2.3, two different driver models that track a cycle within limits, and with the same tracking on average, can give rise to different fuel consumption. Hence, different tracking behavior within specified tracking limits is important, and consequently it is of importance to study how driver models influence the simulation result.

- There is a need to demonstrate that the proposed method is feasible. This means not only velocity profile tracking, but also that other simulated variables are captured. Further, that good properties like simulation speed are not lost.

- Of course it is natural to compare the simulation methods to each other. Due to the ambiguity caused by tracking limits, it is needed with a discussion of performance measures of drive cycle simulation. Also simulation set up effort, and consistency in design space exploration applications should be compared.

3

Inversion of Nonlinear Systems

Inverse dynamic simulation is a powerful possibility for drive cycle simulation as explained in the previous chapter. The basis is to compute generating variables like fuel flow and engine torque when the output, the velocity profile, is given. This means that this is a problem of system inversion, and this chapter will present the theory to be used. The practical use of the theory described in this chapter will be demonstrated on important powertrain models in Chapter 6.

System inversion means that the system has to be inverted such that the inputs can be expressed as a function of the outputs. Various domain specific solutions to this problem have been developed, e.g. in rigid body dynamics, [25, 32, 34]. A review of methods for inverse dynamic simulation of nonlinear systems in aerospace applications is given in [22] where a method based on numerical differentiation followed by algebraic inversion is presented, and an example of that method is presented in [30]. This method is limited in that it uses an Euler approximation of the system derivatives. Another way to perform system inversion is to use a tool for non-causal simulation like e.g. Dymola. The method there is based on a structural manipulation of equations. However, only minimum phase systems are treated and numerical differentiation of the inputs is used [4]. A different possibility is to use the theory of stable inversion of nonlinear systems. The method chosen here is adopted from [7, 19], and the rest of this chapter pre-

sents it by showing how a system of ordinary differential equations is manipulated in order to perform an inverse dynamic simulation of it. The method handles minimum- as well as non-minimum phase systems, and in the next chapter it will be shown how numerical differentiation of the inputs is avoided. A large class of systems that can be simulated in the forward dynamic way can thereby also be simulated in the inverse dynamic way. Section 3.2 presents a new extension to handle conditional functions. The extension is straight forward (and thus perhaps known by others), but is most useful and needed in practical vehicle simulation.

3.1 Stable inversion of non linear systems

Let $u(t)$ be the inputs, $y(t)$ the outputs, and $x(t)$ the states of a given system. Suppose that the system can be written on input-affine form

$$\begin{aligned} \dot{x}(t) &= f(x(t)) + g(x(t))u(t) \\ y(t) &= h(x(t)) \end{aligned} \tag{3.1}$$

and that the number of inputs q equals the number of outputs. As mentioned in the previous chapter, the inverse simulation computes required control inputs from a given drivecycle. For a system such as (3.1), the inverse dynamic simulation problem can be stated as: Given a desired output $y_d(t)$, e.g. a drive cycle, that is smooth, the problem is to solve

$$\begin{aligned} \dot{x}_d(t) &= f(x_d(t)) + g(x_d(t))u_d(t) \\ y_d(t) &= h(x_d(t)) \end{aligned} \tag{3.2}$$

for the input $u_d(t)$ such that (3.2) is satisfied. Possibly it is also interesting to solve for $x_d(t)$, the corresponding state trajectories that produces the desired output. With the notion smooth, it is here meant that a signal is sufficiently many times continuously differentiable. From here on it will be assumed that all functions are smooth such that the necessary derivatives exists and can be computed. It will also be assumed that $f(0) = 0$ and $h(0) = 0$. This can always be achieved for systems like (3.1) by a simple change of coordinates. The basic idea behind the inversion is to use the standard change of coordinates for exact linearization, in order to express the input as a function of the output.

3.1.1 Coordinates transformations for stable inversion

The inversion is based on coordinates transformations, which is described in [7] and [20]. To do this, the notion of relative degree is used. Consider a single-input single-output system of the form (3.1). Let *L* be the standard notation for Lie derivatives according to

$$
\begin{aligned}
L_f^0 h(x) &= h(x) \\
L_f h(x) &= \sum_i f_i(x) \frac{\partial}{\partial x_i} h(x) \\
L_f^r h(x) &= L_f(L_f^{r-1} h(x))
\end{aligned}
$$

where $f_i(x)$ is the i:th row of $f(x)$. The system is said to have relative degree r at a point x^0 if the following holds:

$$
\begin{aligned}
&L_g L_f^k h(x) = 0 \text{ for all } x \text{ in a neighborhood of } x^0 \text{ and all } k < r-1 \\
&L_g L_f^{r-1} h(x^0) \neq 0
\end{aligned}
\tag{3.3}
$$

For systems with multiple outputs and multiple inputs, the vector relative degree $r = (r_1, r_2, \ldots, r_q)$ at a point x^0 is defined as: for all $1 \le j \le q$, for all $1 \le i \le q$, for all $k < r_i - 1$, and for all x in a neighborhood of x^0

$$
L_{g_j} L_f^k h_i(x) = 0 \tag{3.4}
$$

and the $q \times q$ matrix

$$
\tilde{\beta}(x^0) = \begin{bmatrix}
L_{g_1} L_f^{r_1-1} h_1(x^0) & \ldots & L_{g_q} L_f^{r_1-1} h_1(x^0) \\
L_{g_1} L_f^{r_2-1} h_2(x^0) & \ldots & L_{g_q} L_f^{r_2-1} h_2(x^0) \\
\vdots & \vdots & \vdots \\
L_{g_1} L_f^{r_q-1} h_q(x^0) & \ldots & L_{g_q} L_f^{r_q-1} h_q(x^0)
\end{bmatrix} \tag{3.5}
$$

is nonsingular.

To illustrate the interpretation of the notion of relative degree, consider a system with single-input single-output being in a state x^0. Differentiate the output $y(t)$ with respect to time and obtain

$$
\begin{aligned}
y^{(1)}(t) &= \frac{\partial h}{\partial x} \frac{dx}{dt} = \frac{\partial h}{\partial x}(f(x(t)) + g(x(t))u(t)) \\
&= L_f h(x(t)) + L_g h(x(t)) u(t)
\end{aligned}
$$

If the relative degree r is larger than 1 in a neighborhood of x^0, by definition (3.3) it holds that $L_g h(x(t)) = 0$ near x^0 and

$$
y^{(1)}(t) = L_f h(x(t))
$$

If higher orders of the output derivatives are computed we finally get

$$y^{(k)}(t) = L_f^k h(x(t)) \text{ for all } k < r \text{ and in a neighborhood of } x^0$$
$$y^{(r)}(t) = L_f^r h(x^0) + L_g L_f^{r-1} h(x^0) u(t)$$

That is, the relative degree of the system is the number of times one has to differentiate the output $y(t)$ for the input $u(t)$ to appear explicitly. For a multi-input multi-output system there is a corresponding reasoning. Differentiate $y_i(t)$ until at least one $u_j(t)$ appears. This will happen at exactly the r_i:th derivative of $y_i(t)$ due to (3.4).

Example 3.1 Study the system (2.1) extended with a state for the total energy needed to propel the vehicle

$$m\dot{v}(t) = F(t) - mgc_r - \frac{1}{2}\rho c_d A v^2(t)$$
$$\dot{W}(t) = F(t)v(t)$$

Let $x_1 = v$, $x_2 = W$, $u = F$ and $y = v$, then this system can be written in standard statespace form as

$$\dot{x}_1 = -gc_r - \frac{1}{2}\frac{\rho c_d}{m} A x_1^2 + \frac{1}{m}u = f_1(x) + g_1(x)u$$
$$\dot{x}_2 = x_1 u = g_2(x)u$$
$$y = x_1 = h(x)$$

Since the input does not affect the output directly the relative degree $r > 0$. Computing the first Lie-derivative in the direction g gives

$$L_g h(x) = \begin{bmatrix} \frac{\partial h}{\partial x_1} & \frac{\partial h}{\partial x_2} \end{bmatrix} \begin{bmatrix} g_1(x) \\ g_2(x) \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{m} \\ x_1 \end{bmatrix} = \frac{1}{m} \neq 0 \forall x$$

Hence the relative degree of this system is $r = 1$.

The first step in the inversion procedure is to compute the relative degree. The next step is to partially linearize the system. This is done by differentiating $y_i(t)$ until at least one $u_j(t)$ appears explicitly. Define $\xi_k^i(t) = y_i^{(k-1)}(t)$ for $i = 1,\ldots,q$ and $k = 1,\ldots,r_i$ and let

$$\xi(t) = \left(\xi_1^1(t),\xi_2^1(t),\ldots,\xi_{r_1}^1(t),\xi_1^2(t),\ldots,\xi_{r_2}^2(t),\ldots,\xi_{r_q}^q(t)\right)^T$$
$$= \left(y_1(t),\dot{y}_1(t),\ldots,y_1^{(r_1-1)}(t),y_2(t),\ldots,y_2^{(r_2-1)}(t),\ldots,y_q^{(r_q-1)}(t)\right)$$

Now a change of coordinates can be defined. Since y is a function of x, it can be written as

$$\begin{bmatrix}\xi\\ \eta\end{bmatrix} = \psi(x) \tag{3.6}$$

where η are variables needed to fill out the coordinate change. See [20] also for a more thorough discussion on the choice of the new coordinates. With this choice of coordinates, the following representation of the system is achieved

$$\begin{cases} \dot{\xi}_1^i(t) = \xi_2^i(t) \\ \quad\vdots \\ \dot{\xi}_{r_i-1}^i(t) = \xi_{r_i}^i(t) \\ \dot{\xi}_{r_i}^i(t) = \alpha_i(\xi(t),\eta(t)) + \beta_i(\xi(t),\eta(t))u(t) \end{cases} \quad \text{for } i = 1,\ldots,q \tag{3.7a}$$

$$\dot{\eta}(t) = s_a(\xi(t),\eta(t)) + s_b(\xi(t),\eta(t))u(t) \tag{3.7b}$$

which is partially linear in ξ. The vector α and the matrix β are

$$\begin{aligned} \alpha(\xi(t),\eta(t)) &= \begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_q\end{bmatrix} = L_f^r h(\psi^{-1}(\xi(t),\eta(t))) \\ \beta(\xi(t),\eta(t)) &= \begin{bmatrix}\beta_1\\ \vdots\\ \beta_q\end{bmatrix} = L_g L_f^{r-1} h(\psi^{-1}(\xi(t),\eta(t))) \end{aligned} \tag{3.8}$$

The only constraint in the choice of η is that the jacobian matrix of $\psi(x)$ must be nonsingular at x^0 so that it is a local coordinate transformation in a neighborhood of x^0. It is always possible to find such a $\psi(x)$ [20].

Example 3.2 Consider the system in Example 3.1. Since the relative degree of that system was $r = 1$, the first new coordinate is $\xi = y$. To form a complete coordinate change (3.6), the new coordinates has to be filled out. One possible example of such an coordinate change that is also invertible is to choose

$$\begin{bmatrix} \xi \\ \eta \end{bmatrix} = \begin{bmatrix} y \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

which gives the new system

$$\begin{bmatrix} \dot{\xi} \\ \dot{\eta} \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -gc_r - \frac{1}{2}\frac{\rho c_d}{m} A\xi^2 + \frac{1}{m}u \\ \xi u \end{bmatrix} =$$

$$= \begin{bmatrix} L_f h(\psi^{-1}(\xi, \eta)) + L_g h(\psi^{-1}(\xi, \eta))u \\ \xi u \end{bmatrix} = \begin{bmatrix} \alpha(\xi, \eta) + \beta(\xi, \eta)u \\ \xi u \end{bmatrix}$$

Other choices for η could be considered, but in most cases it works well to choose some of the old coordinates which makes it easy to calculate the inverse coordinate change.

Let $y^{(r)}$ denote $y^{(r)} = \begin{bmatrix} y_1^{(r_1)} & \cdots & y_q^{(r_q)} \end{bmatrix}$ Then it can be seen in (3.7a) that $y^{(r)}(t) = \alpha(\xi(t), \eta(t)) + \beta(\xi(t), \eta(t))u(t)$. By the definition of relative degree, $\beta(\xi(t), \eta(t))$ is nonsingular. Given a desired output trajectory $y_d(t)$ the required control input $u_d(t)$ can be calculated as

$$u_d(t) = \beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t))) \tag{3.9}$$

Here it is seen that in order to calculate the required input $u_d(t)$ for the system to follow the prescribed trajectory, not only $y_d(t)$ has to be known, but also the trajectories of $\eta(t)$ and the r first derivatives of $y_d(t)$. The trajectories $\eta(t)$ here represents the systems internal dynamics, or zero dynamics. If the state trajectories $x_d(t)$ producing the desired output is of interest they can be calculated from the inverse coordinate change

$$x(t) = \psi^{-1}(\xi(t), \eta(t))$$

Example 3.3 Consider the system in Example 3.2. It is seen in (3.9) that if the trajectories of $\eta(t)$, $\xi(t)$, and $\dot{\xi}(t)$, i.e., $\eta(t)$, $y(t)$, and $\dot{y}(t)$, are known, the required input $u(t)$ can be calculated as in (3.9)

$$u(t) = \beta(\xi(t), \eta(t))^{-1}(y^{(r)}(t) - \alpha(\xi(t), \eta(t))) =$$
$$= m\left(\dot{y}(t) + gc_r + \frac{1}{2}\frac{\rho c_d}{m} A y^2\right)$$

Note that for this particular system there is no need to know the trajectory of the zero dynamics in order to calculate the required input.

3.1.2 Finding the trajectories of the zero dynamics

As can be seen in (3.9) both the output trajectory and the zero dynamics are needed to compute the required input. How the zero dynamics are to be solved depends on the system studied. There are three main classes of systems that can be written on the form (3.7a), (3.7b). The first class is all systems where the relative degree equals the dimension of the system, which means that there are no zero dynamics. The second class is all systems with stable zero dynamics, i.e., minimum phase systems, and the third class is the systems with unstable zero dynamics, i.e., non-minimum phase systems.

For systems without zero dynamics, (3.9) becomes a system of static equations where the required input can be calculated from the desired output and its derivatives. In the case of stable zero dynamics the procedure is also straight forward. Substitute (3.9) in (3.7b),

$$\begin{aligned}\dot{\eta}(t) &= s_a(\xi(t), \eta(t)) + \\ &+ s_b(\xi(t), \eta(t))(\beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t)))) \\ &\equiv s(\eta(t), Y_d(t))\end{aligned} \tag{3.10}$$

where

$$Y_d(t) = \left(y_1(t), \dot{y}_1(t), \ldots, y_1^{(r_1)}(t), y_2(t), \ldots, y_2^{(r_2)}(t), \ldots, y_q^{(r_q)}(t)\right)$$

Choose appropriate initial values for $\eta(t)$, and solve the system of differential equations, (3.10), in order to find the trajectories of the zero dynamics. Since $\eta(t)$ is a function of $x(t)$ the initial values can be chosen from knowledge of the physics. In the case of unstable zero dynamics however, there is no such straight forward way, since then (3.10) can not be integrated

as an initial value problem. It is not possible to solve the unstable zero dynamics in the general case, but in drive cycle simulations the whole desired output trajectory is known before hand. This gives the possibility of computing a non causal solution of (3.10) and still receive a stable result.

Example 3.4 Consider Example 3.2. Using (3.9) in the zero dynamics gives

$$\dot{\eta}(t) = \xi(t)u(t) = \xi(t)(\beta(\xi(t),\eta(t))^{-1}(y^{(r)}(t) - \alpha(\xi(t),\eta(t)))) =$$
$$= y(t)m\left(\dot{y}(t) + gc_r + \frac{1}{2}\frac{\rho c_d}{m}Ay^2\right)$$

Since $\dot{\eta}$ does not depend on η, the zero dynamics is stable and can be integrated as an initial value problem.

Unstable zero dynamics. Inversion of a system with unstable zero dynamics puts more restrictions on what kind of nonlinearities that can be handled than with stable zero dynamics. To explain the thoughts behind the inversion algorithm, the ideas will be presented on linear systems. A more thorough description is given in [7] and [19].

Consider a system

$$\dot{\eta}(t) = A\eta(t) + Bu(t); \qquad \eta(\pm\infty) = 0 \tag{3.11}$$

where A has no eigenvalues on the imaginary axis. For such systems it is possible to find a similarity transformation $\eta = T\tilde{\eta}$, where T is invertible, that brings the system on a form where

$$\begin{bmatrix}\dot{\tilde{\eta}}_1(t)\\ \dot{\tilde{\eta}}_2(t)\end{bmatrix} = \begin{bmatrix}\tilde{A}_n & 0\\ 0 & \tilde{A}_p\end{bmatrix}\begin{bmatrix}\tilde{\eta}_1(t)\\ \tilde{\eta}_{2(t)}\end{bmatrix} + \begin{bmatrix}\tilde{B}_1\\ \tilde{B}_2\end{bmatrix}u(t); \qquad \eta(\pm\infty) = 0 \tag{3.12}$$

where $\tilde{A}_n$ has all eigenvalues in the left half plane and $\tilde{A}_p$ has all eigenvalues in the right half plane. The solution to the boundary value problem (3.12) is

$$\tilde{\eta}(t) = \int_{-\infty}^{\infty}\phi(t-\tau)\tilde{B}u(\tau)d\tau$$

where

$$\phi(t) = \begin{bmatrix}1(t)e^{\tilde{A}_n t} & 0\\ 0 & -1(-t)e^{\tilde{A}_p t}\end{bmatrix}$$

and $1(t)$ is the unit step function. Solving for the first states $\eta_1(t)$ in (3.12) is easy. The system is stable and the solution

$$\begin{aligned}\tilde{\eta}_1(t) &= \int_{-\infty}^{\infty} 1(t-\tau)e^{\tilde{A}_n(t-\tau)}\tilde{B}_1 u(\tau)d\tau \\ &= \int_{0}^{\infty} e^{\tilde{A}_n\tau}\tilde{B}_1 u(t-\tau)d\tau\end{aligned}$$

can be computed with numerical solvers for ODE:s, such as Euler or Runge-Kutta methods. If the input signal $u(t)$ is known before hand, the solution to the last states $\eta_2(t)$ can be computed in a stable way as

$$\begin{aligned}\tilde{\eta}_2(t) &= \int_{-\infty}^{\infty} -1(\tau-t)e^{\tilde{A}_p(t-\tau)}\tilde{B}_2 u(\tau)d\tau \\ &= \int_{-\infty}^{0} -e^{\tilde{A}_p\tau}\tilde{B}_2 u(t-\tau)d\tau\end{aligned}$$

i.e. a non causal simulation can be used to obtain a numerical solution. Hence a stable simulation of the system can be done.

For a nonlinear system such as (3.1), a separation of the system in a stable and unstable part as in (3.12) can not be done in general. In [7], a Picard-like iteration method that is based on linearization of the nonlinear system is used for finding the states $\eta(t)$ in (3.10). Theory about Picard iterations is given in [16]. First, the zero dynamics is written as

$$s(\eta(t), Y_d(t)) = A\eta + (s(\eta(t), Y_d(t)) - A\eta)$$

which can be interpreted such as that the nonlinearities is seen as disturbances to a linear system. A is a linear approximation of $s(\cdot)$, typically $\left.\frac{\partial s}{\partial \eta}\right|_{(0,\,0)}$. The iteration method simulates the linear system with the disturbances (nonlinearities) as inputs according to

$$\eta_{m+1}(t) = \int_{-\infty}^{\infty} \phi(t-\tau)[s(\eta_m(\tau), Y_d(\tau)) - A\eta_m(\tau)]d\tau \qquad (3.13)$$

where $\phi(t)$ is the state transition matrix for the linear system. If a change of coordinates that brings the linearization on the form of (3.12) is applied, the system becomes

$$\dot{\tilde{\eta}}(t) = T^{-1}s(T\tilde{\eta}(t), Y_d(t)) =$$
$$= T^{-1}AT\tilde{\eta}(t) + T^{-1}(s(T\tilde{\eta}(t), Y_d(t)) - AT\tilde{\eta}(t))$$

and the iteration (3.13) can be separated on the form (3.12)

$$\begin{bmatrix} \tilde{\eta}_{1,\,m+1}(t) \\ \tilde{\eta}_{2,\,m+1}(t) \end{bmatrix} = \int_{-\infty}^{\infty} \phi(t-\tau)T^{-1}(s(T\tilde{\eta}_m(\tau), Y_d(\tau)) - AT\tilde{\eta}_m(\tau))d\tau \quad (3.14)$$

where

$$\phi(t) = \begin{bmatrix} 1(t)e^{\tilde{A}_n(t)} & 0 \\ 0 & -1(-t)e^{\tilde{A}_p(t)} \end{bmatrix}$$

For this iteration to converge, $s(\eta, Y_d)$ can not be too nonlinear. In [7] it is shown that the iteration converges to a unique solution $\eta(t)$ that satisfies (3.10) if $s(\eta, Y_d)$ satisfies the local Lipschitz condition

$$\begin{aligned} &\|[s(x(t), u(t)) - Ax(t)] - [s(y(t), v(t)) - Ay(t)]\|_1 \\ &\quad \le K_1\|x(t) - y(t)\|_1 + K_2\|u(t) - v(t)\|_1 \end{aligned} \quad (3.15)$$

in an r neighborhood, i.e. $x(t), u(t), v(t)$ and $y(t)$ with $\|\cdot\|_\infty$ norms less than r, where K_1 and K_2 is positive real constants fulfilling

$$\begin{aligned} K_1 &< \frac{1}{\|\phi(\cdot)\|_\alpha} \\ \|Y_d(\cdot)\|_1 K_2 &< \frac{r(1 - \|\phi(\cdot)\|_\alpha)K_1}{\|\phi(\cdot)\|_\beta} \end{aligned} \quad (3.16)$$

where $\|Y_d(\cdot)\|_1 + \|Y_d(\cdot)\|_\infty < r$ and the following definition is used

$$\|\phi(\cdot)\|_\alpha = \sum_j \max_k \|\phi_{j,\,k}(\cdot)\|_1 \quad (3.17)$$

$$\|\phi(\cdot)\|_\beta = n\sup_\tau \max_{j,\,k} |\phi_{j,\,k}(\tau)| \quad (3.18)$$

where n is the dimension of ϕ.

A similar iteration method based on linearization of the nonlinear system is presented in Chapter 8 in [2].

3.1.3 Remarks and extensions

The most intuitive way to model a system is with a forward dynamic model as (2.2). It has been shown that a large subclass of systems like (2.2) can, in a methodical way, be transformed into a corresponding inverse dynamic model. Further relaxations can be made in order to extend the class of invertible systems.

The first restriction that was made in the previous section was that the system had to be written in input affine form (3.1). This is not a big restriction. Most physical systems are on this form or can easily be rewritten on this form. The restriction could also be relaxed such that $g(x)u$ is relaxed into $g(x, u)$ as long as it is invertible in u.

Secondly, the system may be time varying, i.e. explicitly dependant on time t, e.g. due to parameter variations. Since the minimum phase case is only about solving IVPs and algebraic equations, parameter variations are easily taken care of. The system (3.1) in time dependant form is

$$\begin{aligned} \dot{x}(t) &= f(x(t), t) + g(x(t), t)u(t) \\ y(t) &= h(x(t), t) \end{aligned}$$

which is easily handled by an ODE solver. The same argument is valid for the zero dynamics. In the non minimum phase case it is however more complex to handle parameter variations and not even possible for all cases, see [8, 9].

3.2 Handling of conditional functions

In the forward dynamic models it is easy to specify logical conditions on the system as e.g.

$$\begin{aligned} \dot{x}(t) &= \begin{cases} f_1(x(t), u(t)), \text{ if } x, u \in S_1 \\ f_2(x(t), u(t)), \text{ if } x, u \in S_2 \end{cases} \\ y(t) &= h(x(t), u(t)) \end{aligned} \tag{3.19}$$

Such conditions are easy to handle in numerical ODE solvers for initial value problems, see e.g. [2]. Typically the system is simulated as an ODE between switch points. An algorithm searches for the switches and makes sure that a simulation step is taken at the switch point. The states just before the switch are taken as initial values to the system that is in effect after the switch, and the simulation continues from there.

In inverse dynamic simulation, i.e. evaluating (3.9),

$$u_d(t) = \beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t))) \tag{3.20}$$

there may potentially be a problem. The reason is that in order to compute α and β it is required to compute the inverse coordinate change, ψ^{-1}, as is seen in (3.8)

$$\alpha(\xi(t), \eta(t)) = L_f^r h(\psi^{-1}(\xi(t), \eta(t)))$$
$$\beta(\xi(t), \eta(t)) = L_g L_f^{r-1} h(\psi^{-1}(\xi(t), \eta(t)))$$

On one hand, if a logical condition appears in the coordinate change as e.g.

$$\begin{bmatrix} \xi \\ \eta \end{bmatrix} = \begin{cases} \psi_1(x(t)), \text{ if } x \in X_1 \\ \psi_2(x(t)), \text{ if } x \in X_2 \end{cases}$$

there is no way to know which coordinate change of $\psi_1(x)$ or $\psi_2(x)$ to use when computing the inverse coordinate change. Since the inverse coordinates change is needed in (3.20), this kind of condition can make the system non invertible. However, note that sometimes, by choosing η the problem can be avoided, as illustrated in the examples in Chapter 6. For the non minimum phase case it may be necessary also to consider derivatives of η see e.g. [8, 9] when and how such cases can be handled.

On the other hand, it is worth noting, see (3.20), that in order to find the solution to the inverse dynamic problem there is no function that has to be inverted, except the inverse coordinate change. This means that nonlinear functions in general does not pose any problem other than stated above. Rather, the functions in the right hand side of (3.20)$_1$ are evaluated for the current value of ξ and η. Recall specifically that β^{-1} does not denote the inverse function, but instead the inverted value.

3.3 Summary

It has been shown how non linear systems are inverted by use of a coordinate change. The inversion algorithm can be summarized as follows:

1 Find the relative degree of the system.

2 Rewrite the system by a coordinate change $\begin{bmatrix} \xi^T & \eta^T \end{bmatrix}^T = \psi(x)$ on the form

$$\begin{cases} \dot{\xi}_1^i(t) = \xi_2^i(t) \\ \quad\vdots \\ \dot{\xi}_{r_i-1}^i(t) = \xi_{r_i}^i(t) \\ \dot{\xi}_{r_i}^i(t) = \alpha_i(\xi(t), \eta(t)) + \beta_i(\xi(t), \eta(t))u(t) \end{cases} \quad \text{for } i = 1, \ldots, q \tag{3.21}$$

$$\dot{\eta}(t) = s_a(\xi(t), \eta(t)) + s_b(\xi(t), \eta(t))u(t) \tag{3.22}$$

3 Possibly, to compute α and β, the inverse coordinate change has to be computed

4 Solve the zero dynamics (3.22) by substituting u with (3.9).

5 Use (3.9), i.e. $u_d(t) = \beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t)))$ to compute the required input.

The values for the output and its derivatives, $y_d^{(r)}$ and ξ, are given. Steps 1 and 2 are performed off line. Step 3 requiring ψ^{-1} is solved analytically if possible, or tabulated off line, or solved numerically online. Steps 4 and 5 are performed on line.

4

Extending Inverse Simulation and Driver Models

In drive cycle simulations the vehicle has to follow a prescribed speed profile within certain limits, as described in Chapter 2. To achieve such good drive cycle tracking, more or less sophisticated driver models are needed. Driver models for both forward and inverse dynamic simulation are here discussed, and for this it is recalled that exact tracking is physically impossible. The new implicit driver model presented in Section 4.2 together with the inversion procedure presented in Chapter 3 forms a new concept of inverse dynamic vehicle simulation. To compare tracking and behavior, the different driver models are in the following tested on the New European Drive Cycle, NEDC, EEC Directive 90/C81/01, see Figure 2.1 and the Federal Test Procedure FTP75, see Figure 2.2.

4.1 Tracking in forward dynamic simulation

In forward dynamic simulation a driver model acts as a controller to track the reference speed trajectory. As can be seen in Figure 2.6 the driver model compares the drivecycle with the actual speed and decides how to control the inputs, e.g., accelerator pedal, brake pedal, clutch, and gear selector, in a way such that the vehicle follows the prescribed drive cycle within defined limits. Mathematically, the driver model is an operator $D_f(v_c, v)$ that for each time t_a defines a mapping from the drivecycle

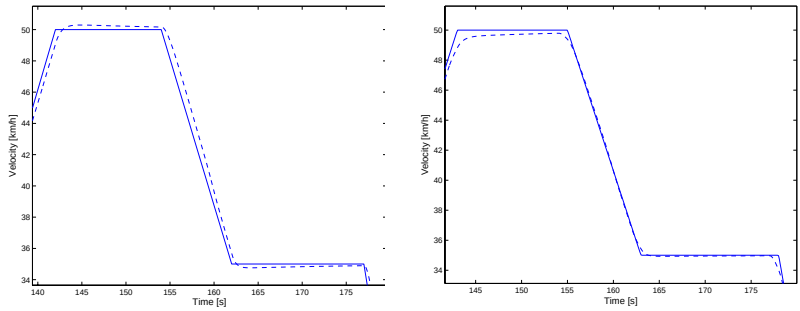

Figure 4.1 Driver behavior in forward dynamic simulation of a part of NEDC. To the left is the behavior of a driver model implemented as a PI-controller depicted. To the right a PI-controller together with a proportional look ahead of 1 s is used.

$v_c \in \{v_c(t) | t_{start} \le t \le t_{stop}\}$ and the actual speed $v \in \{v(t) | t_{start} \le t \le t_a\}$ to control the input to the vehicle. A driver model can be as simple as a PID-controller [10] or a more sophisticated controller with prediction abilities [21]. For the PID-controller the driver operator is

$$D_f(v_c, v) = K_p(v_c(t_a) - v(t_a)) + K_i \int_{t_{start}}^{t_a} (v_c(s) - v(s))ds + \\ + K_d \frac{d}{dt}(v_c(t_a) - v(t_a)) \tag{4.1}$$

This simple driver model does not have look ahead, since it uses only information of past and present values of the drive cycle. The parameters of the driver model determines driver alertness, which in this type of simulation results in tightness of cycle tracking. For examples of tracking behavior for forward driver models, see Figure 4.1.

4.2 Enabling Tracking in inverse dynamic simulation

In the inverse dynamic simulation, the reference speed trajectory is followed exactly if the physics of the model allows it. As mentioned in [11], and as can be seen in Equation (3.9), the reference speed trajectory has to be continuously differentiable as many times as the relative degree of the system, see also [20]. If it is not sufficiently differentiable, then the trajectory has to be smoothed to create a reference speed that the simulated model can follow with bounded states and inputs [11]. This smoothing of the drive cycle will here be given the interpretation of an implicit driver model. The shape of the manipulated (smooth) speed profile decides the driver behavior. As in the

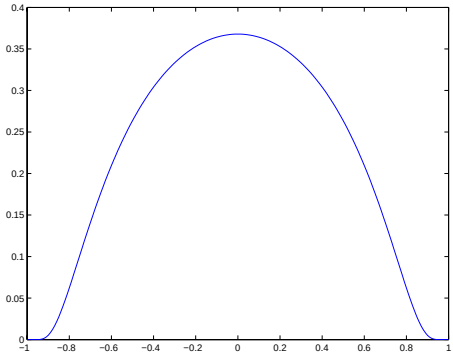

Figure 4.2 The convolution kernel, Equation (4.4), used to smooth the drivecycle. (a=1)

forward dynamic simulation a driver operator, $D_i v_c$, can be defined that maps the drivecycle $v_c \in \{v_c(t) | t_{start} \le t \le t_{stop}\}$ to a desired speed profile $v_d \in \{v_d(t) | t_{start} \le t \le t_{stop}\}$ that is to be followed by the vehicle. There are several possibilities to smooth the drive cycle. One way is to filter the drive cycle with a linear low pass filter with relative degree of at least the same order as the system. This will ensure that the filtered drive cycle is differentiable sufficiently many times for Equation (3.9) to be solvable, see [20]. Using a non causal linear filter gives the driver look ahead properties, for an example see Figure 4.3. Another way to smooth the drive cycle is to use the driver operator

$$D_i v_c = \frac{1}{C} \int_{-\infty}^{\infty} g(t-\tau) v_c(\tau) d\tau \tag{4.2}$$

to compute the desired trajectory $v_d(t)$ as the convolution

$$v_d(t) = \frac{1}{C} \int_{-\infty}^{\infty} g(t-\tau) v_c(\tau) d\tau \tag{4.3}$$

A good choice for the convolution kernel, $g(t)$, see Figure 4.2, is to use the definition

$$g(t) = \begin{cases} e^{\frac{-a^2}{a^2 - t^2}}, & |t| \le a; \\ 0, & \text{otherwise;} \end{cases} \tag{4.4}$$

where the parameter a is a tracking time constant. A smaller a gives "tighter" tracking which corresponds to a more aggressive or alert driver behavior. See Figure 4.3 and Figure 4.4 for examples of drive cycle smoothing.

Since the choice (4.4) of the convolution kernel $g(t)$ has compact support and is infinitely many times continuously differentiable, the resulting trajectory $v_d(t)$ is infinitely many times continuously differentiable. Moreover, the calculation of the derivatives does not require numerical differentiation. Instead the derivatives are calculated using the following convolution

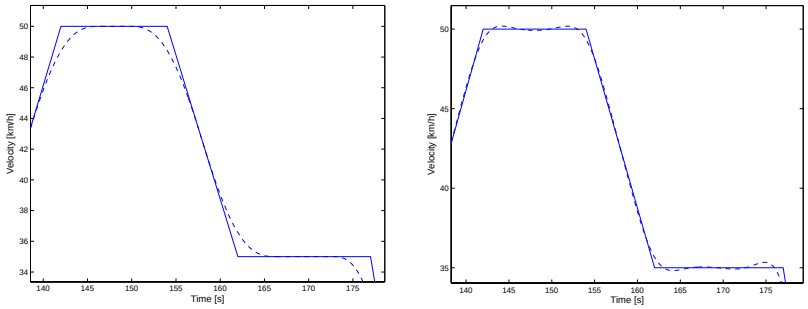

Figure 4.3 Tracking behavior for inverse dynamic simulation on part of the ECE cycle. To the left the behavior of the tracking defined by Equation (4.3) and (4.4) with a tracking time constant of 5 seconds is shown. To the right a non causal linear low pass filter is used. In both cases the tracking has look ahead.

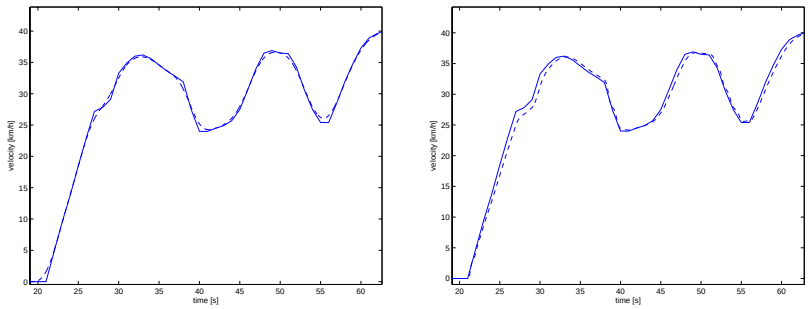

Figure 4.4 Tracking behavior for inverse dynamic (left) and forward dynamic (right) simulation on part of the FTP75 cycle. The solid line is the drive cycle and the dashed line is simulated velocity. In the inverse dynamic simulation the tracking is defined by Equation (4.3) and (4.4) with a tracking time constant of 2 seconds. The forward dynamic driver model is defined by (4.1).

$$v_d^{(r)}(t) = \frac{1}{C}\int_{-\infty}^{\infty} g^{(r)}(t-\tau)v_c(\tau)d\tau$$

Linear filtering of the drive cycle gives only asymptotically exact tracking while the use of (4.4), due to compact support, gives exact tracking on large parts of the drive cycle.

4.3 New extended inverse simulation

Using the driver model presented in the previous section, and the inversion technique described in the previous chapter, the quasi-static inverse simulation is extended to inverse dynamic simulation as depicted in Figure 4.5. The shaping of the drive cycle is interpreted as an implicit driver model. The inverse drivers tracking behavior is specified in the velocity domain and is independent of the vehicle that it is going to control. In fact, this way of specifying tracking behavior is closer to human behavior when performing drive cycle tests on a chassis dynamometer. In that case the driver is not specified by controller parameters, but rather in terms of tracking smoothness.

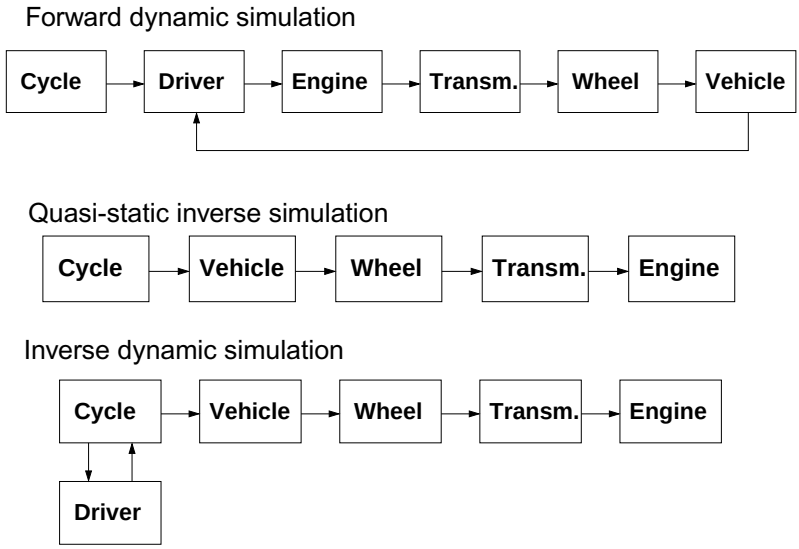

Figure 4.5 Schematic depiction of simulation methods. On the top is forward dynamic, then quasi-static inverse simulation as in Advisor and the QSS-Toolbox, and on the bottom the new extended inverse dynamic concept that is an extension of quasi-static inverse simulation. In inverse dynamic simulation simulated states are calculated back-wards from the drive cycle. The drive cycle tracking is explicit, whereas the driver model is implicit. In standard forward dynamic simulation the driver model is explicit whereas tracking is implicit.

The new simulation concept is a bridge that ties forward dynamic simulations as in, e.g., PSAT, to quasi-static inverse simulations as in, e.g., Advisor, in the sense that a common model data base can be used. Instead of specifying dedicated inverse models as those used in, e.g. Advisor the models can be specified in the same way as in forward dynamic simulation. By use of symbolic math tools, the systematic inversion procedure can be more or less automatized.

5

Usability of Simulations and Performance Measures

In Chapters 3 and 4 a new method was proposed for drive cycle simulation. Before that, in Chapter 2 the usage of such drive cycle simulations were discussed, and requirements were stated. To verify that the new method fulfills these demands it has to be tested accordingly, and it also has to be compared to existing methods. The purpose of this chapter is to discuss the foundations for these test simulations. The actual evaluation will take place in Chapter 7 after having presented the relevant vehicle models in Chapter 6.

The requirements are both of subjective and objective character and this will be elaborated in this chapter. A first observation is that there are some phenomena that may lead to inconclusive simulation results if they are not treated carefully, and these will be treated in Sections 5.1 and 5.2. With these sections as a background, but mainly based on Chapter 2, then Section 5.3 will give the simulation measures to be used in Chapter 7.

5.1 Issues related to drive cycle tracking

As can be seen in Chapter 4, tracking in forward and inverse dynamic simulation is achieved in different ways. This leads to different properties of forward and inverse dynamic simulation, and a comparison is not straight forward. To illustrate how driver behavior can influence simulation result,

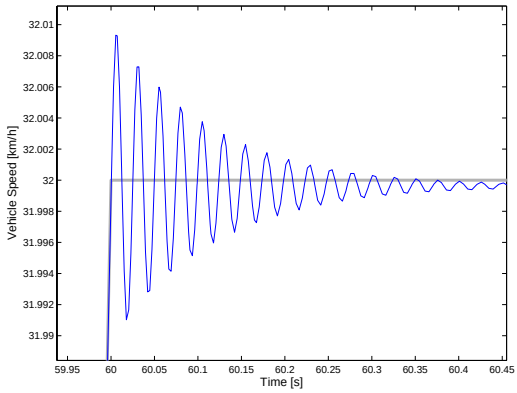

Figure 5.1 Tracking behavior on part of NEDC of a badly tuned driver model. The thick line is the drive cycle and the thin line is the tracking of a driver model.

part of a drive cycle simulation is presented in Figure 5.1. The driver model there tracks the drive cycle almost correct on average, but the oscillating behavior gives higher fuel consumption than a smoother tracking would give. Clearly it is not enough to only verify that tracking is within limits, but also the behavior within limits has to be studied.

When parameters in a model are changed it is possible that the tracking behavior can change from smooth to oscillating like in the example. While the tracking is the same on average this change can be fatal for the simulation results such as fuel consumption. Of course, this is also an important factor in design space exploration treated in the next section.

5.2 Design space exploration

As stated in Section 2.2, design space exploration is an important area of usage for drive cycle simulations. Design space exploration means that drive cycle simulations are repeated many times where one or several parameters vary between each simulation. For example, one way to choose the gear ratios in the transmission that gives minimal fuel consumption is to let the simulation be a part of an optimization loop, where the simulation is used to calculate the objective function. This requires a batch of simulations where the gear ratios are varied between every simulation. Other cases are optimization of engine size for minimal fuel consumption and tuning of control system parameters. For simulation experiments such as those mentioned it is important, according to the discussion in the previous section, that the

Simulation Method	Driver model	Fuel cons [l/100km] r=0.3m	Fuel cons [l/100km] r=0.45m
Forward	$K_p=1.5\cdot10^{-4}$ $K_i=1\cdot10^{-5}$	8.619	7.315
Forward	$K_p=5\cdot10^{-4}$ $K_i=5\cdot10^{-6}$	8.634	7.293
Inverse	a=2	8.595	7.265
Inverse	a=1	8.627	7.271

Table 5.1: Simulation of the powertrain model from Section 6.1 with a change in wheel radius from 0.3m to 0.45m. The driver models are according to Chapter 4. For the inverse dynamic simulation the relative order in fuel consumption between the three driver models is the same for both wheel radius. This is not the case for the forward dynamic simulations.

drive cycle is followed, for every simulation, not only on average but also exhibits the same characteristics within tracking limits.

To exemplify that the problem can easily occur, a simple driveline is used in a few repeated simulation experiments. The model chosen here is the model that is going to be presented in Section 6.1 and it is simulated to study the fuel consumption in the NEDC driving cycle. The fuel consumption is taken as the integral of the maximum of the fuel flow given by the model and an idle flow. Simulations are first performed both with the inverse dynamic method and the forward dynamic method. Keeping the same driver models, the simulation is performed again but with the wheel radius changed from $0.3m$ to $0.45m$. The results are shown in Table 5.1. For the forward case the order between the two cases are reversed ($8.619 < 8.634$ compared to $7.315 > 7.293$), which means that the order in fuel consumption between the driver models is not preserved when wheel radius is changed. This indicates that it can be difficult to evaluate simulations where model parameters change if the tracking characteristics is not preserved. For the inverse case the order is consistent ($8.595 < 8.627$ and $7.265 < 7.271$).

5.3 Evaluations of simulations

With the examples presented above in mind, we now turn to the task of trying to find fair comparison measures that can be used to compare simulation methods with each other.

Recall Figure 4.5 for the computational scheme of inverse dynamic simulation and how it differs from forward dynamic simulation. A first objective is to demonstrate feasibility of the inverse dynamic method, and this will be done in Chapter 7. For simulation comparisons, both subjective and objective measures are presented in the following.

5.3.1 Setup effort

The first thing that has to be done when simulating a vehicle is to implement the equations describing the vehicle in a simulation tool, e.g., Matlab/Simulink. This might demand that the equations are rewritten in order to suit the chosen simulation tool. The next thing to do is to design a driver model that is capable of controlling the vehicle such that good enough drive cycle tracking is achieved. How to implement the vehicle and driver model differs depending on simulation method and hence the first measure is:

Simulation measure 1: Simulation setup effort. *Given a set of differential equations describing a physical system. How easy is it to implement those equations in a simulation tool? How easy is it to design a driver model for the system studied?*

5.3.2 Simulation time

When running sets of simulations, as in e.g. design space exploration, or when simulation is part of an optimization loop, simulation speed is very important. For example, lets say that one wants to try out five different gear ratios on five gears each. To find the optimal ratios it is likely that more ratios have to be tried out, but, already this small example requires $5^5 = 3125$ simulations. This shows clearly that simulation speed is very important and it calls for the second measure:

Simulation measure 2: Simulation time. *The time it takes to simulate a given model tracking a given drive cycle on a given computer.*

5.3.3 Result dependency on parameters

As mentioned in Section 5.2, when running batches of simulations, i.e. sets of simulations, it is important that the simulation result is consistent between simulations. For example, if a gear ratio is changed it is likely that, using the same driver model, the tracking of the drive cycle also will change. It is not always easy to verify that a change in fuel consumption, after a change in gear ratio, stems from the parameter change and not the different drive cycle tracking that it might give. From the examples it is seen that it is important that the simulation method preserves the relative order in the dependency between parameters and simulation result.

Simulation measure 3: Preservation of result dependency on parameters. *How the simulation method preserves the dependency between parameter changes and simulation result.*

Using the performance measures presented, inverse dynamic simulation will be compared to forward dynamic simulation in Chapter 7, but first some powertrain models that are going to be used in the comparison will be presented.

6

Powertrain Applications

As mentioned before, drive cycle simulation is important to study fuel consumption and emissions. To increase precision or to model emissions, it is necessary to capture the behavior in transients that are dominating in the processes generating fuel consumption and emissions. Therefore it is necessary to include the dominant dynamics, and not only use quasi-static simulation. In this chapter three of the most important dynamics that capture such transients will be presented. The first is the dynamics of the intake manifold pressure which is modeled in Section 6.1 and added to the vehicle model. Another important property is resonances in the driveline mainly caused by torsion in the drive shafts. This is modeled in Section 6.2. For diesel engines with exhaust gas recycling the gas flow around the engine is important for emission modeling. This flow is modeled in Section 6.3.

Thus, the models presented in this chapter are extensions to the quasi-static models used in tools such as Advisor [33] or the QSS-toolbox [15]. However, they can all be treated using the inversion procedure from Chapter 3. This will be detailed in the coming subsections, and they will be used for comparison of different simulation methods in Chapter 7.

Even though the examples in this chapter are motivated from the application they each impose interesting and illustrative mathematical characteristics. The first example with intake manifold pressure illustrate the extension with one extra state compared to quasi-static modelling. The second example with drive shaft torsion includes a resonant system and zero dynamics, and

the third example with gas flow around diesel engines also includes non-minimum phase, i.e. unstable zero dynamics, that can not be simulated in object oriented tools such as Dymola [4].

6.1 Capturing engine dynamics - a model with additional state

This example model is chosen to be as simple as possible but still including interesting engine dynamics that can not be modelled in the framework used in tools such as Advisor or the QSS-toolbox. It consists of a simple stiff drive line with a dynamic engine model where the intake manifold pressure is a state, see Figure 6.1. The models presented in this section are standard as presented in e.g. [18, 21] and follow the notation there. See also Chapter 9 for notation.

6.1.1 Forward dynamic model

The rotating parts of the engine is modelled as a rotating mass

$$J_e \dot{\omega}_e = T_{cr} - T_e \tag{6.1}$$

where T_{cr} is the driving torque on the crankshaft and T_e is the torque from the driveline on the flywheel. The driving torque, T_{cr}, is here modeled from a standard mean value engine model.

$$BMEP = IMEP_g - FMEP - PMEP \tag{6.2}$$

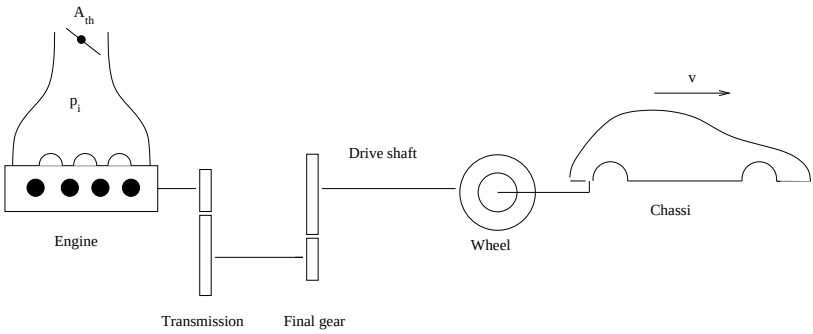

Figure 6.1 A simple powertrain model with a dynamic engine model. The input is throttle area and the output is vehicle speed.

where the brake-, indicated-, friction-, and pump mean effective pressure are

$$BMEP = 4\pi\frac{T_{cr}}{V_d} \tag{6.3}$$

$$IMEP_g = \frac{m_f q_{hv}\eta_{th}}{V_d} \tag{6.4}$$

$$FMEP = \left(\left(0.464 + 0.0072\left(\frac{S\omega_e}{\pi}\right)^{1.8}\right)\pi_{bl}\cdot 10^5 + 0.0215\cdot\frac{4\pi T_{cr}}{V_d}\right)\cdot$$

$$\cdot\,\xi_{aux}\sqrt{\frac{0.075}{B}} \tag{6.5}$$

$$PMEP = p_e - p_i \tag{6.6}$$

The friction model (6.5) is according to [29]. The engine is assumed to run stoichiometric

$$\frac{m_{ac}}{m_f} = \left(\frac{A}{F}\right)_s\lambda \tag{6.7}$$

and the normalized air mass charge is modelled according to [17] as

$$m_{ac} = \eta_{vol}p_i\frac{V_d}{RT_i} = (s_i p_i - y_i)\frac{V_d}{RT_i} \tag{6.8}$$

The intake manifold is modelled with a dynamic model with intake manifold pressure as state

$$\dot{p}_i = \frac{RT_i}{V_i}\left(\dot{m}_{at} - \frac{\omega_e}{2\pi n_r}m_{ac}\right) \tag{6.9}$$

The throttle is modeled as in [18]

$$\dot{m}_{at} = \frac{p_a}{\sqrt{RT_a}}A_{eff}\Psi(p_r) \tag{6.10}$$

where the pressure ratio is $p_r = p_i/p_a$, the effective throttle area is $A_{eff} = A_{th}C$ and

$$\Psi(p_r) = \begin{cases} \sqrt{\frac{2\gamma}{\gamma-1}\left(p_r^{\frac{2}{\gamma}} - p_r^{\frac{\gamma+1}{\gamma}}\right)}, & \text{if } p_r > \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}} \\ \sqrt{\gamma\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}}, & \text{otherwise} \end{cases} \tag{6.11}$$

The transmission is modeled as an ideal gear with constant efficiency

$$\begin{aligned} \omega_e &= i\omega_w \\ T_e i\eta_{tm} &= T_w \end{aligned} \tag{6.12}$$

The wheel is modeled as a stiff rolling wheel without slip

$$\begin{aligned} T_w &= Fr \\ \omega_w r &= v \end{aligned} \tag{6.13}$$

The force equation for the vehicle is modeled with a quadratic air resistance and constant rolling resistance as

$$F = m\dot{v} + \frac{1}{2}c_d\rho A v^2 + mgc_r \tag{6.14}$$

Combining Equations (6.1)-(6.14) and assuming constant ambient temperature and pressure gives the state space representation

$$\begin{aligned} \dot{v}(t) &= c_1 v^2(t) + c_2 v^{1.8}(t) + c_3 p_i(t) + c_4 \\ \dot{p}_i(t) &= c_5 v(t) + c_6 v(t) p_i(t) + c_7 \Psi\left(\frac{p_i(t)}{p_a}\right) A_{eff}(t) \end{aligned} \tag{6.15}$$

where $A_{eff}(t)$ is considered as input to the model, and c_i is the lumped model parameters. As can be seen, the model has two states, intake manifold pressure and vehicle speed. Note also that the system contains an if-statement in the definition of $\Psi(p_r)$ in (6.11), i.e., the system is on the form (3.19).

6.1.2 Inverse dynamic model

Since the system (6.15) is on input affine form there is a good chance that it is invertible. To use the notation of Chapter 3 the system is rewritten with $x_1(t) = v(t)$, $x_2(t) = p_i(t)$, $u(t) = A_{eff}(t)$ and $y(t) = v(t)$, which gives

$$\dot{x}_1(t) = c_1 x_1^2(t) + c_2 x_1^{1.8}(t) + c_3 x_2(t) + c_4 = f_1(x(t))$$

$$\dot{x}_2(t) = c_5 x_1(t) + c_6 x_1(t) x_2(t) + c_7 \Psi\left(\frac{x_2(t)}{p_a}\right) u(t) =$$

$$= f_2(x(t)) + g_2(x(t)) u(t)$$

$$y(t) = x_1(t) = h(x(t))$$

First the relative degree of the system has to be found. This is done using the definition (3.3). The Lie derivatives are

$$L_g h(x) \equiv 0, \forall x$$

$$L_g L_f h(x) = c_3 c_7 \Psi\left(\frac{x_2(t)}{p_a}\right) \neq 0, \forall x \in \{x | (x_2 \neq p_a)\}$$

which means that the relative degree is $r = 2$ for all points except when $x_2 = p_a$ for which a relative degree can not be defined. The engine studied here are naturally aspirated which means that the point where the relative degree is undefined, i.e. ambient pressure equals intake manifold pressure, will never be reached. Since the relative degree is $r = 2$, the same as the system order, there are no zero dynamics. The coordinate change (3.6) becomes

$$\begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} x_1 \\ c_1 x_1^2 + c_2 x_1^{1.8} + c_3 x_2 + c_4 \end{bmatrix} \tag{6.16}$$

where $\xi_1 = y$ and $\xi_2 = \dot{y}$. The inverse coordinate change, $x = \psi^{-1}(\xi)$, is the solution to the system of equations (6.16), which has the solution

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \xi_1 \\ \frac{1}{c_3}(\xi_2 - c_1 \xi_1^2 - c_2 \xi_1^{1.8} - c_4) \end{bmatrix} \tag{6.17}$$

Note that the conditional if-statement in (6.11) in the model does not appear in the inverse coordinate change. Now $\alpha(\xi(t))$ and $\beta(\xi(t))$ can be computed according to (3.8) which results in

$$\begin{aligned}\alpha(\psi(x)) &= (2c_1x_1 + 1.8c_2x_1^{0.8})(c_1x_1^2 + c_2x_1^{1.8} + c_3x_2 + c_4) + \\ &\quad + c_3(c_5x_1 + c_6x_1x_2) \\ \beta(\psi(x)) &= c_3c_7\Psi\left(\frac{x_2}{p_a}\right)\end{aligned} \tag{6.18}$$

Given a desired trajectory $y_d(t)$ and its two first derivatives $\dot{y}_d(t)$ and $\ddot{y}_d(t)$, the required input $u_d(t)$ to produce that output can be calculated according to (3.9) as

$$u_d(t) = \beta(\psi(x))^{-1}(y_d^{(r)}(t) - \alpha(\psi(x))) \tag{6.19}$$

where $x(t)$ is calculated from (6.17). Since the system does not have any zero dynamics it is sufficient to solve a set of algebraic equations, (6.17) and (6.19), to calculate the required input. Note that, as stated in Section 3.2, non linear functions such as $\Psi(\cdot)$ does not have to be inverted, rather it is evaluated for the current values of the states and then only the computed value of $\beta(\psi(x))$ is inverted.

6.2 Powertrain with driveline dynamics - a model with zero dynamics

The second example is a driveline with a drive shaft flexibility added between the transmission (6.12) and the wheel (6.13), [23, 24]. The system described in Section 6.1 had a relative degree that equaled the order of the system, but this system has a relative degree that is lower than the system order, i.e., it is a system with zero dynamics.

6.2.1 Forward dynamic model

The torque in the drive shaft is modelled as spring and damper

$$T_d = k_{ds}\theta_d + c_{ds}\dot{\theta}_d \tag{6.20}$$

Equation (6.12) now becomes

$$\begin{aligned}\omega_e &= i\omega_d \\ T_e i\eta_{tm} &= T_d\end{aligned} \tag{6.21}$$

and the relation between the rotational speeds on each side of the driveshaft is

$$\dot{\theta}_d = \omega_d - \omega_w \tag{6.22}$$

The wheel torque in (6.13) is

$$T_w = T_d \tag{6.23}$$

The system that is going to be studied is the same as in the previous section modified as described by (6.20)-(6.23). With the introduction of the driveshaft flexibility, two states are added to the system. Since the engine and vehicle speed becomes decoupled the first of the new states is engine speed defined by (6.1). The second extra state is the torsion of the driveshaft defined by (6.22). Collecting all equations and setting $x_1 = v$, $x_2 = \omega_e$, $x_3 = \theta_d$, $x_4 = p_i$, $u = A_{eff}$ and $y = v$ gives the following state space system

$$\begin{aligned}
\dot{x}_1(t) &= c_1 x_1^2(t) + c_2 x_1(t) + c_3 x_2(t) + c_4 x_3(t) + c_5 = f_1(x(t)) \\
\dot{x}_2(t) &= c_6 x_1(t) + c_7 x_2(t) + c_8 x_2^{1.8}(t) + c_9 x_3(t) + c_{10} x_4(t) + c_{11} = \\
&= f_2(x(t)) \\
\dot{x}_3(t) &= c_{12} x_1(t) + c_{13} x_2(t) = f_3(x(t)) \\
\dot{x}_4(t) &= c_{14} x_2(t) + c_{15} x_2(t) x_4(t) + c_{16} \Psi\left(\frac{x_4(t)}{p_a}\right) u(t) = \\
&= f_4(x(t)) + g_4(x(t)) u(t) \\
y(t) &= x_1(t) = h(x(t))
\end{aligned} \tag{6.24}$$

6.2.2 Inverse dynamic model

To find the inverse model first the Lie-derivatives are computed which gives

$$\begin{aligned}
L_g h(x) &= L_g L_f h(x) \equiv 0, \forall x \\
L_g L_f^2 h((x)) &= c_3 c_{10} c_{16} \Psi\left(\frac{x_4}{p_a}\right) \neq 0, \forall x \in \{x | (x_4 \neq p_a)\}
\end{aligned} \tag{6.25}$$

hence the relative degree is $r = 3$ for all points except when $x_4 = p_a$ for which a relative degree can not be defined. A possible coordinate change is

$$
\begin{aligned}
\xi_1 &= x_1 \\
\xi_2 &= \dot{x}_1 = c_1 x_1^2 + c_2 x_1 + c_3 x_2 + c_4 x_3 + c_5 \\
\xi_3 &= \ddot{x}_1 = (2c_1 x_1 + c_2)(c_1 x_1^2 + c_2 x_1 + c_3 x_2 + c_4 x_3 + c_5) + \\
&\quad + c_3(c_6 x_1 + c_7 x_2 + c_8 x_2^{1.8} + c_9 x_3 + c_{10} x_4 + c_{11}) + \\
&\quad + c_4(c_{12} x_1 + c_{13} x_2) \\
\eta &= x_4
\end{aligned}
\tag{6.26}
$$

There is a freedom to choose the zero dynamics here. Since $\dot{x}_4$ contains an if-statement the coordinate change is chosen so that the if-statement is included in the zero dynamics, see Section 3.2. Since the zero dynamics comes from the spring-damper model of the drive shaft it is stable and can be solved with a forward dynamic simulation, where if-statements are no problem. If the if-statement would appear in some of the equations for ξ, it is not easy to know if the if- or else-part of the if-statement should be used in the inverse coordinate change.

Because of the type of nonlinearities, as seen in (6.26), the inverse coordinate change can not be solved analytically. It has to be solved numerically, using, e.g., a Newton type method or fixed point iteration.

Given the coordinate change (6.26), α and β can be computed as in (3.8), and the system (6.24) can be written as

$$
\begin{aligned}
\dot{\xi}_1(t) &= \xi_2(t) \\
\dot{\xi}_2(t) &= \xi_3(t) \\
\dot{\xi}_3(t) &= \alpha(\xi(t), \eta(t)) + \beta(\xi(t), \eta(t))u(t) \\
\dot{\eta}(t) &= f_4(\psi^{-1}(\xi(t), \eta(t))) + g_4(\psi^{-1}(\xi(t), \eta(t)))u(t)
\end{aligned}
\tag{6.27}
$$

$\beta = L_g L_f^2 h$ is given in (6.25) and α is given by

$$\alpha = L_f^3 h = ((2c_1x_1 + c_2)c_4 + c_3c_9)(c_{12}x_1 + c_{13}x_2) + c_3c_{10}c_{14}x_2 +$$
$$(2c_1(c_1x_1^2 + c_2x_1 + c_3x_2 + c_4x_3 + c_5) + (2c_1x_1 + c_2)^2 + c_3c_6 + c_4c_{12}) \cdot$$
$$(c_1x_1^2 + c_2x_1 + c_3x_2 + c_4x_3 + c_5) + c_3c_{10}c_{15}x_4$$
$$+ ((2c_1x_1 + c_2)c_3 + c_3(c_7 + 1.8c_8x_2^{0.8}) + c_4c_{13}) \cdot$$
$$(c_6x_1 + c_7x_2 + c_8x_2^{1.8} + c_9x_3 + c_{10}x_4 + c_{11}) +$$
$$+ ((2c_1x_1 + c_2)c_4 + c_3c_9)(c_{12}x_1 + c_{13}x_2) + c_3c_{10}c_{14}x_2 + c_{15}x_4$$

If now (3.9) is inserted in the last row of (6.27) the zero dynamics becomes

$$\dot{\eta}(t) = f_4(\psi^{-1}(\xi(t), \eta(t))) +$$
$$+ g_4(\psi^{-1}(\xi(t), \eta(t)))(\beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t))))$$

The zero dynamics is solved numerically with a standard ODE solver. When $\eta(t)$ is known the required input $u_d(t)$ is calculated according as in (3.9)

$$u_d(t) = \beta(\xi(t), \eta(t))^{-1}(y_d^{(r)}(t) - \alpha(\xi(t), \eta(t)))$$

As stated in Section 3.2 and as in the previous example, note that non linear functions such as $\Psi(\cdot)$ do not have to be inverted.

6.3 Gas flow control of a diesel engine - a non-minimum phase example

The model in this section is a simplified version of the model presented by Wahlström [31], and describes the air flow around a diesel engine with EGR (exhaust gas recycling) and VGT (variable geometry turbine), see Figure 6.2. Control variables are the EGR valve area and the are of the VGT. These two inputs are used to control the mass flow and its composition into the cylinder. Here the fuel flow and engine speed are considered as model parameters. Outputs are normalized air fuel ratio, λ, and EGR ratio, x_{egr}. It will later be shown that the system is a non minimum phase system for many operating points. Intuitively it can be reasoned as: When the VGT is closed, initially, the flow past the turbine will decrease, which results in a decrease of compressor flow and a decrease in λ. However, there will be a build up of pressure in the exhaust manifold, which after a while will result in a higher flow past the turbine, which speeds up the turbo, and after a while results in an increase of fresh air flow, which means that λ increases above its initial value.

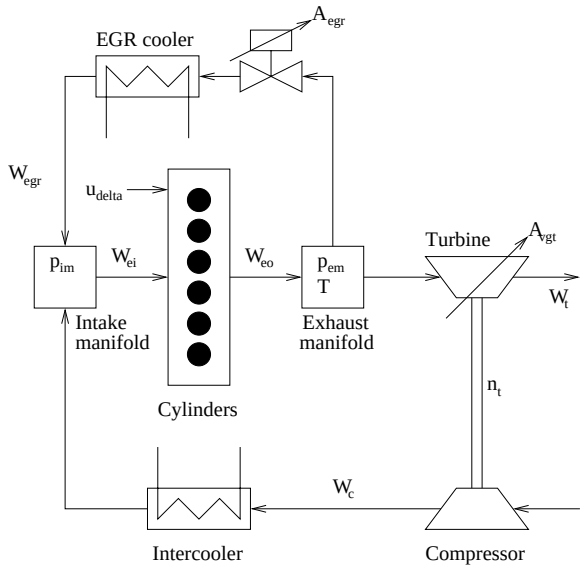

Figure 6.2 A schematic picture of a diesel engine with EGR and VGT. The inputs are the areas of the EGR-valve and the VGT. Outputs are the EGR ratio and the normalized air fuel ratio. The figure is adapted from [31].

6.3.1 Forward dynamic model

The intake manifold is modeled as a dynamic system as

$$\dot{p}_i = \frac{R_a T_i}{V_i}(W_c + W_{egr} - W_{ei}) \tag{6.28}$$

Likewise the exhaust manifold is modeled as

$$\dot{p}_e = \frac{R_e T_{em}}{V_e}(W_{eo} - W_t - W_{egr}) \tag{6.29}$$

The air/EGR mass flow into the cylinders is modeled as

$$W_{ei} = \frac{\eta_{vol} p_i N V_d}{n_r R_a T_i} \tag{6.30}$$

The fuel mass flow into the cylinders is

$$W_f = \frac{N}{n_r} n_{cyl} u_\delta \tag{6.31}$$

and the mass flow out of the engine is

$$W_{eo} = W_f + W_{ei} \tag{6.32}$$

The normalized air fuel ratio is

$$\lambda = \frac{W_c}{W_f \left(\frac{A}{F}\right)_s} \tag{6.33}$$

The EGR system consist of a pipe connecting the exhaust manifold to the intake manifold. In this pipe there is a throttle to control the flow of EGR. The flow is modeled as

$$W_{egr} = \frac{A_{egr} p_e \Psi_{egr}\left(\frac{p_i}{p_e}\right)}{\sqrt{T_{em} R_e}} \tag{6.34}$$

where

$$\Psi_{egr}\left(\frac{p_i}{p_e}\right) = \begin{cases} \sqrt{\frac{2\gamma_e}{\gamma_e - 1}\left(\left(\frac{2}{\gamma_e + 1}\right)^{\frac{2}{\gamma_e - 1}} - \left(\frac{2}{\gamma_e + 1}\right)^{\frac{\gamma_e + 1}{\gamma_e - 1}}\right)}, & \text{if } \frac{p_i}{p_e} < \left(\frac{2}{\gamma_e + 1}\right)^{\frac{\gamma_e}{\gamma_e - 1}} \\ \sqrt{\frac{2\gamma_e}{\gamma_e - 1}\left(\left(\frac{p_i}{p_e}\right)^{\frac{2}{\gamma_e}} - \left(\frac{p_i}{p_e}\right)^{\frac{\gamma_e + 1}{\gamma_e}}\right)}, & \text{if } \left(\frac{2}{\gamma_e + 1}\right)^{\frac{\gamma_e}{\gamma_e - 1}} \le \frac{p_i}{p_e} \le 1 \\ 0, & \text{if } 1 < \frac{p_i}{p_e} \end{cases} \tag{6.35}$$

The throttle is modeled as a first order linear dynamic system as

$$\dot{A}_{egr} = \frac{1}{\tau_{egr}}(-A_{egr} + u_{egr}) \tag{6.36}$$

The EGR ratio is defined as

$$x_{egr} = \frac{W_{egr}}{W_c + W_{egr}} \tag{6.37}$$

The turbo consist of a turbine and a compressor connected with a stiff shaft. The speed dynamics of the turbo is given by

$$\dot{\omega}_t = \frac{1}{J_t}(M_t - M_c) \tag{6.38}$$

where the driving torque from the turbine is

$$M_t = \eta_m \frac{W_t c_{pe} \Delta T_t}{\omega_t} \tag{6.39}$$

and the braking torque from the compressor is

$$M_c = \frac{W_c c_{pa} \Delta T_c}{\omega_t} \tag{6.40}$$

A lumped model for the mechanical efficiency and the temperature difference of the turbine is used according to

$$\eta_m \Delta T_t = T_{em} \eta_{tm} \left(1 - \left(\frac{p_a}{p_e} \right)^{\frac{\gamma_e - 1}{\gamma_e}} \right) \tag{6.41}$$

The turbine mass flow is modeled as

$$W_t = \frac{p_e}{\sqrt{T_{em}}} \sqrt{1 - \left(\frac{p_a}{p_e} \right)^{K_t}} A_{vgt} \tag{6.42}$$

The compressor mass flow is given by

$$W_c = \frac{p_a \pi R_c^3 \omega_t}{R_a T_a} \Phi_c \tag{6.43}$$

and the temperature difference over the compressor is given by

$$\Delta T_c = \frac{T_a}{\eta_c}\left(\left(\frac{p_i}{p_a}\right)^{\frac{\gamma_a - 1}{\gamma_a}} - 1\right) \tag{6.44}$$

Combining Equations (6.28) - (6.44) results in the following state space description

$$\dot{p}_i = c_1 p_i + c_2 \omega_t + c_3 A_{egr} p_e \Psi\left(\frac{p_i}{p_e}\right)$$

$$\dot{p}_e = c_4 p_i + c_5 A_{egr} p_e \Psi\left(\frac{p_i}{p_e}\right) + c_6 + c_7 p_e \sqrt{1 - \frac{c_8}{p_e^{c_9}}} A_{vgt}$$

$$\dot{\omega}_t = c_{10} p_i^{c_{11}} + c_{12} + \left(c_{13} + \frac{c_{14}}{p_e^{c_{15}}}\right)\sqrt{1 - \frac{c_8}{p_e^{c_9}}} \frac{p_e}{\omega_t} A_{vgt}$$

$$\dot{A}_{egr} = c_{16} A_{egr} - c_{16} u_{egr}$$

with the outputs

$$\lambda = c_{17} \omega_t$$

$$x_{egr} = \frac{c_{18} A_{egr} p_e \Psi\left(\frac{p_i}{p_e}\right)}{c_{19}\omega_t + c_{18} A_{egr} p_e \Psi\left(\frac{p_i}{p_e}\right)}$$

As was mentioned in Chapter 3, it is assumed that $f(0) = 0$ and $h(0) = 0$ in (3.2). To put the system in that form a coordinate change $p_i = x_1 + c_{20}$, $p_e = x_2 + c_{21}$, $\omega_t = x_3 + c_{22}$, $A_{egr} = x_4 + c_{23}$, $u_{egr} = u_1 + c_{24}$ and $A_{vgt} = u_2 + c_{25}$ with appropriate coefficients is applied which results in

$$\dot{x}_1 = c_1(x_1 + c_{20}) + c_2(x_3 + c_{22}) + \\ + c_3(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\frac{(x_1 + c_{20})}{(x_2 + c_{21})}\right) = f_1(x)$$

$$\dot{x}_2 = c_4(x_1 + c_{20}) + c_5(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\frac{(x_1 + c_{20})}{(x_2 + c_{21})}\right) + c_6 + \\ + c_7(x_2 + c_{21})\sqrt{1 - \frac{c_8}{(x_2 + c_{21})^{c_9}}}(u_2 + c_{25}) = f_2(x) + g_2(x)u \quad (6.45)$$

$$\dot{x}_3 = c_{10}(x_1 + c_{20})^{c_{11}} + c_{12} + \\ + \left(c_{13} + \frac{c_{14}}{(x_2 + c_{21})^{c_{15}}}\right)\sqrt{1 - \frac{c_8}{(x_2 + c_{21})^{c_9}}}\frac{(x_2 + c_{21})}{(x_3 + c_{22})}(u_2 + c_{25}) = \\ = f_3(x) + g_3(x)u$$

$$\dot{x}_4 = c_{16}(x_4 + c_{23}) - c_{16}(u_1 + c_{24}) = f_4(x) + g_4(x)u$$

If a similar manipulation is done with the outputs, $\lambda = y_1 - c_{25}$ and $x_{egr} = y_2 - c_{26}$, they become

$$y_1 = c_{17}(x_3 + c_{22}) + c_{25} = h_1(x)$$

$$y_2 = \frac{c_{18}(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\frac{(x_1 + c_{20})}{(x_2 + c_{21})}\right)}{c_{19}(x_3 + c_{22}) + c_{18}(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\frac{(x_1 + c_{20})}{(x_2 + c_{21})}\right)} + \\ + c_{26} = h_2(x) \quad (6.46)$$

6.3.2 Inverse dynamic model

Computing Lie-derivatives in the directions f and g for the system (6.45) - (6.46) gives

$$\begin{bmatrix} L_{g_1}h_1(x) = 0 & L_{g_2}h_1(x) \neq 0 \\ L_{g_1}h_2(x) \neq 0 & L_{g_2}h_2(x) \neq 0 \end{bmatrix}$$

almost everywhere. This means that $\tilde{\beta}(x)$, see (3.5), is nonsingular and hence the system has a vector relative degree $r = \begin{bmatrix} r_1 & r_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \end{bmatrix}$ almost everywhere. Since the original system (6.45) - (6.46) has order four, the

inverse system will have zero dynamics of order two. There are several possibilities to choose the coordinate change (3.6). One choice for $(\xi^T, \eta^T)^T = \psi(x)$ is

$$\begin{bmatrix} \xi_1 \\ \xi_2 \\ \eta_1 \\ \eta_2 \end{bmatrix} = \begin{bmatrix} h_1(x) \\ h_2(x) \\ x_1 \\ x_2 \end{bmatrix} =$$

$$= \begin{bmatrix} c_{17}(x_3 + c_{22}) + c_{25} \\ \dfrac{c_{18}(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\dfrac{(x_1 + c_{20})}{(x_2 + c_{21})}\right)}{c_{19}(x_3 + c_{22}) + c_{18}(x_4 + c_{23})(x_2 + c_{21})\Psi\left(\dfrac{(x_1 + c_{20})}{(x_2 + c_{21})}\right)} + c_{26} \\ x_1 \\ x_2 \end{bmatrix} \tag{6.47}$$

which gives the following inverse system

$$\begin{aligned} \dot{\xi}_1 &= L_f h_1(\psi^{-1}(\xi, \eta)) + L_g h_1(\psi^{-1}(\xi, \eta))u = \alpha_1(\xi, \eta) + \beta_1(\xi, \eta)u \\ \dot{\xi}_2 &= L_f h_2(\psi^{-1}(\xi, \eta)) + L_g h_2(\psi^{-1}(\xi, \eta))u = \alpha_2(\xi, \eta) + \beta_2(\xi, \eta)u \\ \dot{\eta}_1 &= f_1(\psi^{-1}(\xi, \eta)) \\ \dot{\eta}_2 &= f_2(\psi^{-1}(\xi, \eta)) + g_2(\psi^{-1}(\xi, \eta))u \end{aligned}$$

Using (3.9) to insert u, the zero dynamics becomes

$$\begin{aligned} \dot{\eta}_1 &= f_1(\psi^{-1}(\xi, \eta)) \\ \dot{\eta}_2 &= f_2(\psi^{-1}(\xi, \eta)) + \\ &+ g_2(\psi^{-1}(\xi, \eta)) \begin{bmatrix} \beta_1(\xi, \eta) \\ \beta_2(\xi, \eta) \end{bmatrix}^{-1} \left(\begin{bmatrix} \dot{\xi}_1 \\ \dot{\xi}_2 \end{bmatrix} - \begin{bmatrix} \alpha_1(\xi, \eta) \\ \alpha_2(\xi, \eta) \end{bmatrix} \right) \end{aligned} \tag{6.48}$$

Note that because of the choice of the coordinate change (6.47) the if-statement in the function $\Psi((x_1 + c_{20})/(x_2 + c_{21}))$ does not cause any trouble, see Section 3.2. In each simulation step, first η is solved from (6.48) which

gives x_1 and x_2, then the correct part of the if-statement in $\Psi((x_1 + c_{20})/(x_2 + c_{21}))$ is chosen, and the two first equations in (6.47) are solved for x_3 and x_4.

In contrast to the previous example the zero dynamics is in this case unstable, as will now be discussed.

Handling of instabilities in (6.48). In this example the following parameters are used: $c_{20} = 2.06 \cdot 10^5 \text{Pa}$, $c_{21} = 2.41 \cdot 10^5 \text{Pa}$, $c_{22} = 7001 \text{rad/s}$ and $c_{23} = 1 \cdot 10^{-4} \text{m}^2$. Linearize the zero dynamics in (6.48) at the origin

$$\dot{\eta} = A\eta$$

Then the eigenvalues of A are $\sigma_1 = -6.8$ and $\sigma_2 = 15.8$. Clearly the system has unstable zero dynamics in a neighborhood of the origin.

Let $s(\eta_m(\tau), Y_d(\tau))$ be the derivatives of the zero dynamics given by (6.48), i.e., $\dot{\eta} = s(\eta, Y_d)$. Using a diagonalization $\tilde{A} = T^{-1}AT$ of the linearized zero dynamics, e.g. let T be the matrix with the eigenvectors of A, one can simulate the zero dynamics, if it converges, as in (3.14)

$$\begin{bmatrix} \tilde{\eta}_{1,\,m+1}(t) \\ \tilde{\eta}_{2,\,m+1}(t) \end{bmatrix} = \\ = \int_{-\infty}^{\infty} \phi(t-\tau) T^{-1} (s(T\tilde{\eta}_m(\tau), Y_d(\tau)) - AT\tilde{\eta}_m(\tau)) d\tau \tag{6.49}$$

where

$$\phi(t) = \begin{bmatrix} 1(t)e^{-6.8(t)} & 0 \\ 0 & -1(-t)e^{15.8(t)} \end{bmatrix}$$

It is seen, as expected, that one pole is in the right half plane and one pole is in the left half plane. For this system the norms in (3.17) and (3.18) is

$$\|\phi(\cdot)\|_\alpha = \frac{1}{6.8} + \frac{1}{15.8}$$

$$\|\phi(\cdot)\|_\beta = 2$$

According to [7] a solution to the inversion problem can be found that satisfies the system (6.45)-(6.46) if the zero dynamics satisfies (3.15) and (3.16), i.e.,

$$\begin{aligned}&\left\|[s(\eta(t), Y_d(t)) - A\eta(t)] - [s(y(t), V(t)) - Ay(t)]\right\|_1 \\ &\leq K_1\|\eta(t) - y(t)\|_1 + K_2\left\|Y_d(t) - V(t)\right\|_1\end{aligned}$$

and

$$K_1 < \frac{1}{\frac{1}{6.8} + \frac{1}{15.8}}$$

$$\left\|Y_d(\cdot)\right\|_1 K_2 < \frac{r\left(1 - \left(\frac{1}{6.8} + \frac{1}{15.8}\right)\right)K_1}{2}$$

These requirements are fulfilled in a neighborhood of the origin. Staying in this neighborhood, the inverse dynamic simulation is given by (6.49) and (3.9)

$$u_d(t) = \beta(\xi(t), \eta(t))^{-1}\left(\begin{bmatrix}\dot{y}_{1,d}\\ \dot{y}_{2,d}\end{bmatrix} - \alpha(\xi(t), \eta(t))\right) \tag{6.50}$$

7

Simulation Performance and Usability

When performing a simulation there a several properties that are important. As has been mentioned, see Figure 2.8, the ambition has been to combine the advantages of forward dynamic simulation and quasi-static inverse simulation in a new method, inverse dynamic simulation. The solution was proposed in Chapters 3 and 4. Guided by the problem formulation in Section 2.5, performance measures were discussed in Chapter 5. Using these measures inverse dynamic simulation will here be compared to forward dynamic simulation. For this comparison the powertrain models from Chapter 6 will be used.

The first step is to demonstrate feasibility of the inverse dynamic method. As stated in Section 2.5, it is important to be able to include engine and driveline dynamics. Feasibility of inverse dynamic simulation for such models are demonstrated in the simulations in Sections 7.3 and 7.4. Feasibility for non-minimum phase systems is discussed around simulations of the diesel engine model in Section 7.5. Using the simulations of the three presented models, results on simulation time are given in Section 7.6. Section 7.7 is a summarizing discussion.

7.1 Simulations

The simulations of the powertrain models performed in this chapter use parameter values according to Chapter 9. Regarding driver models, the forward model is controlled with a driver model implemented as a PID-controller as in (4.1) and connected to the vehicle model as depicted in Figure 2.6. The PID-parameters are $K_p = 8 \cdot 10^{-5}$, $K_i = 1 \cdot 10^{-6}$ and $K_d = 0$. The inverse dynamic simulation, see Figure 4.5, uses a driver model as described by (4.2) - (4.4) with the parameter $a = 5$. The design of the driver models has been done as follows: First an inverse dynamic driver model has been designed that has a suitable tracking behavior. Then, a forward dynamic PID-type driver model has been designed such that it has comparable response time as the inverse driver model. The models are simulated in the New European Drive Cycle, Figure 2.1. The models does not describe hard decelerations and standstill since no clutch or brakes are included. However, the models are still well suited for comparison between inverse dynamic and forward dynamic simulation.

7.2 Feasibility of the inverse dynamic simulation method

To illustrate that the inverse dynamic simulation method produces valid results, all models from Chapter 6 will be simulated in the following sections. For completeness not only the inputs and outputs, but also all states for all models, and both forward and inverse dynamic simulation, will be presented. First, the model of a stiff driveline from Section 6.1 is studied and the simulations are presented in Figures 7.1 - 7.6. These simulations are also used for the discussion on setup effort in Section 7.3. Second, the model of the flexible driveline from Section 6.2 is studied and the simulations are presented in Figures 7.7 - 7.12. These simulations are also used for the discussion of result dependency on parameters in Section 7.4. Third the diesel engine gas flow model from Section 6.3 is studied and the simulations are presented in Figures 7.13 - 7.20. These simulations are also used for discussion about handling of non-minimum phase systems.

The conclusion from all these is that inverse dynamic simulation gives realistic values not only for the velocity that is tracked, but also for all other variables. Off course this was expected from theory, but it shows that inverse dynamic simulation is not sensitive to numerical approximations and works well.

7.3 Simulation setup effort

In the first comparison between inverse dynamic simulation and forward dynamic simulation, the basic powertrain model from Section 6.1 is used. Both the forward model (6.15) and the inverse dynamic model (6.18) and

(6.19) are simulated in the New European Drive Cycle using parameter values and driver models as stated in Section 7.1. Simulations where carried out in Matlab/Simulink with standard setup of tolerances and solver, i.e., the “ode45” solver with a relative tolerance of 0.001.

7.3.1 Choice of integration step

First some observations on the choice of integration step are made where both automatic step size and manual step size control are commented. For automatic step size control, the result from the first 200 s is shown in Figures 7.1 - 7.3. The inverse model (6.19) is algebraic, which means that there is no integration error no matter what the step length is. The reason for the long step size in the inverse dynamic simulation, Figures 7.1 - 7.3, is the automatic step size control in Simulink, because the step length is chosen by default in Simulink to the simulation time divided by 50, and since no integration error ever occurs the step length is never changed. Note that depending on the purpose of the simulation this may be quite viable and in such a case extremely fast.

To better resolve the signals and to get more similar trajectories between the forward and inverse dynamic simulation, the same simulations are performed again but now with a maximum step length of 1 second, see Figures 7.4 - 7.6. From the simulations performed it is seen that now both the inverse and forward dynamic simulation produces qualitatively the same result.

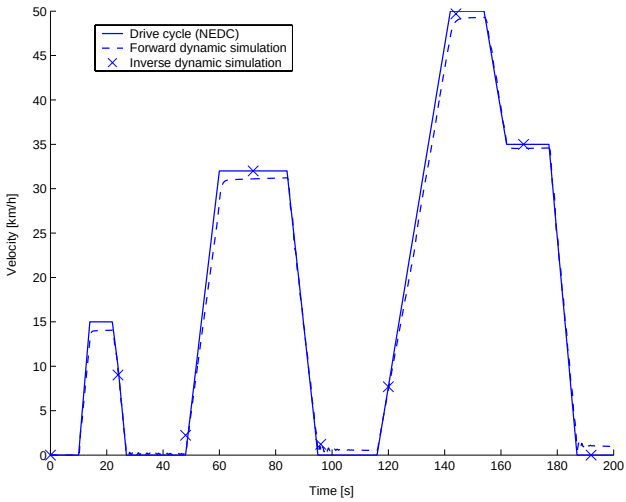

Figure 7.1 Velocity of a simulation of the basic powertrain model from Section 6.1 during the first 200 s of the New European Drive Cycle NEDC.

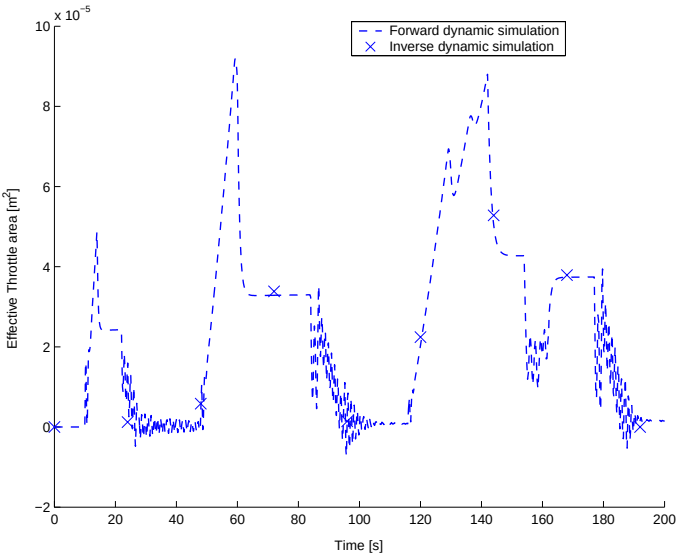

Figure 7.2 Effective throttle area of a simulation of the basic powertrain model from Section 6.1 during the first 200 s of the New European Drive Cycle NEDC.

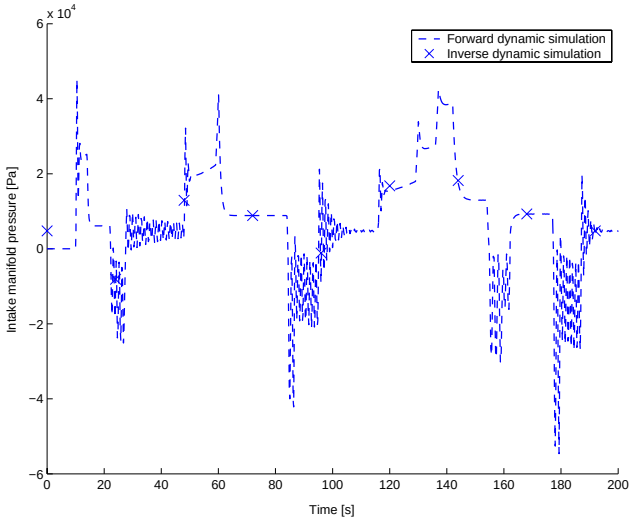

Figure 7.3 Intake manifold pressure of a simulation of the basic powertrain model from Section 6.1 during the first 200 s of the New European Drive Cycle NEDC.

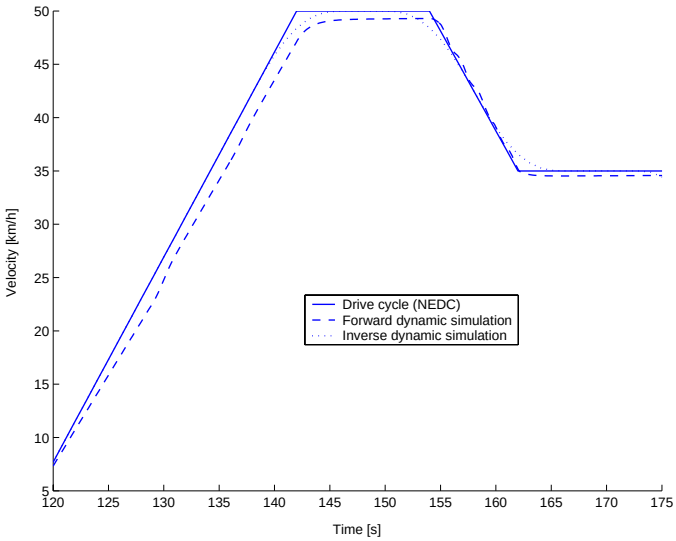

Figure 7.4 Velocity of a simulation of the basic powertrain model from Section 6.1 during a part of NEDC with maximum steplength set to 1 s.

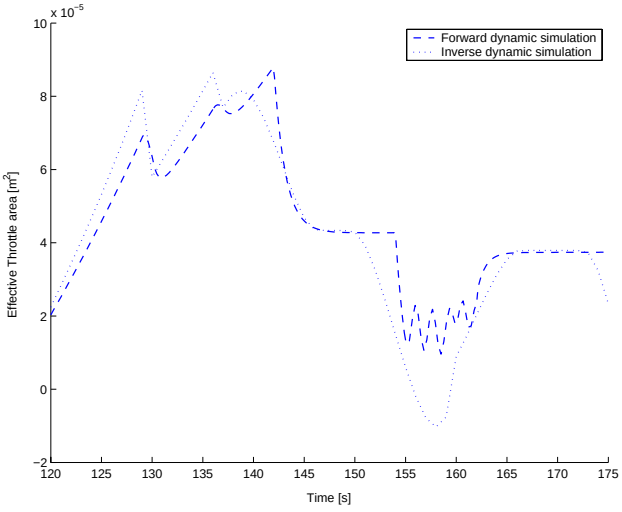

Figure 7.5 Effective throttle area of a simulation of the basic powertrain model from Section 6.1 during a part of NEDC with maximum steplength set to 1 s.

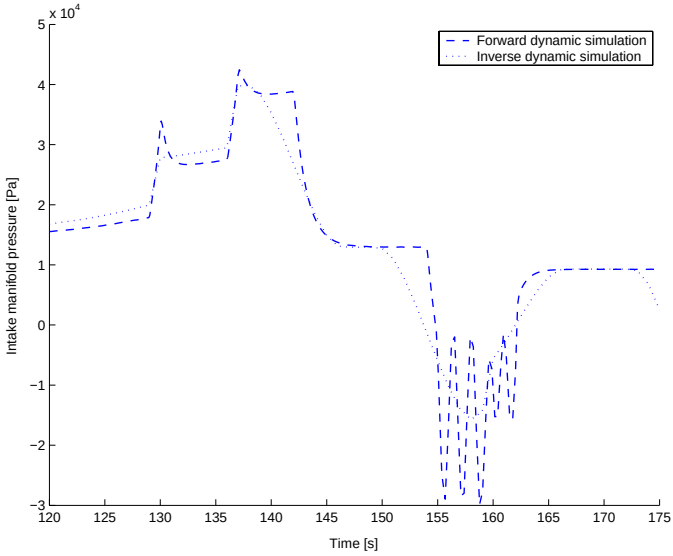

Figure 7.6 Intake manifold pressure of a simulation of the basic powertrain model from Section 6.1 during a part of NEDC with maximum steplength set to 1 s.

7.3.2 Complexity of simulation setup

Above it was seen that both the inverse and forward dynamic simulation produced qualitatively the same result. However, it can be seen in Figures 7.5 and 7.6 that the forward dynamic simulation has an oscillating behavior in throttle position and intake manifold pressure. This stems from the driver model, since for this non linear system it is not sufficient with a simple PID-controller. However, it may take quite some effort to design a driver model that acts as a real driver and adapts to the vehicles behavior in different operating points. On the other hand, it is seen that the inverse dynamic simulation has a smooth control that better resembles the behavior of a real driver. The complexity of designing a driver model may be a drawback for the forward dynamic simulation considering setup effort.

It is straight forward to implement a forward dynamic model in a simulation tool like Matlab/Simulink, whereas an inverse dynamic simulation requires manipulations with the differential equations describing the system before it can be simulated. These manipulations are however systematic as described in Chapter 3, and can be more or less automatized. In the forward dynamic simulation, a drawback is the need to design a realistic driver model, while it is easy to do it in inverse dynamic simulation.

7.4 Result dependency on parameters

As mentioned before, drive cycle simulation can be used for design space exploration like optimizing gear ratios of the transmission, optimizing the size of the engine for minimal fuel consumption, or tuning of control system parameters. Then, it is important that it is easy to run a set of simulations with similar trackings, where some parameters vary between simulations.

To study dependencies on vehicle model parameters, the model from Section 6.2 will be used. A first simulation of a 1700 kg vehicle with a 2.3 liter engine is shown in Figures 7.7 - 7.11. There it is seen that both the forward and inverse dynamic simulation produces qualitatively comparable results, once again confirming the feasibility of inverse dynamic simulation regarding all state variables even in this resonant case. Keeping the driver parameters, the engine size is changed to 1.6 liters and the vehicle mass is changed to 1100kg.

The important effect of the parameter changes can be seen in Figure 7.12. The tracking behavior in the forward simulations is different in the two cases but exactly the same in the inverse dynamic simulations. Note that even though all simulations are within tracking limits, the different behavior within limits can give unwanted effects e.g. on fuel consumption. Note specifically the small oscillations in the velocity for the light weight vehicle

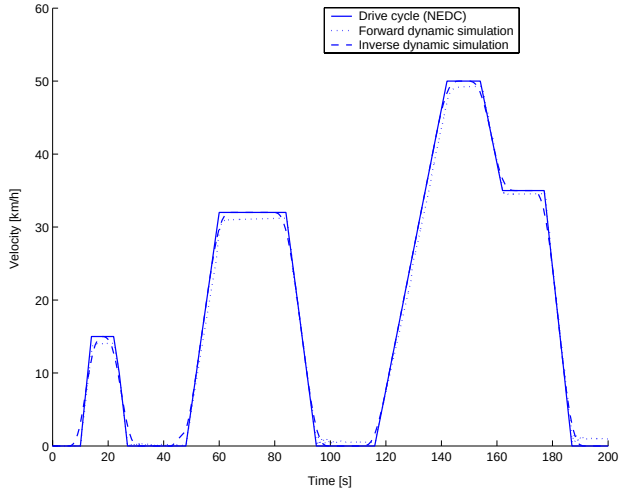

Figure 7.7 Velocity of a simulation of the powertrain model with flexible drive shaft from Section 6.2 during a part of NEDC.

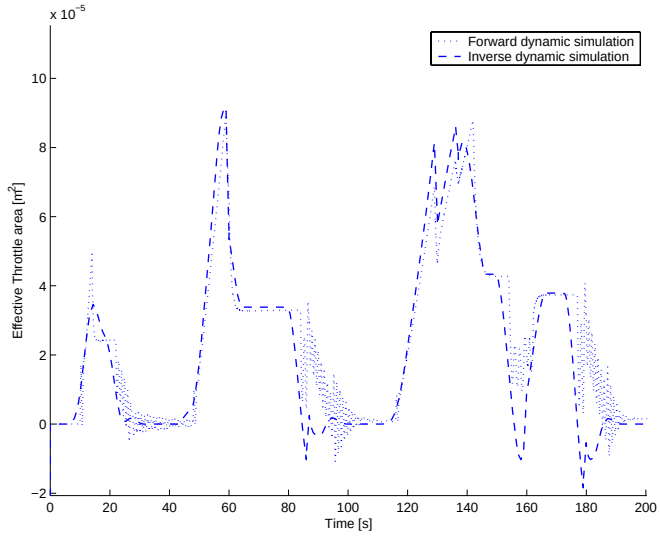

Figure 7.8 Effective throttle area of a simulation of the powertrain model with flexible drive shaft from Section 6.2 during a part of NEDC.

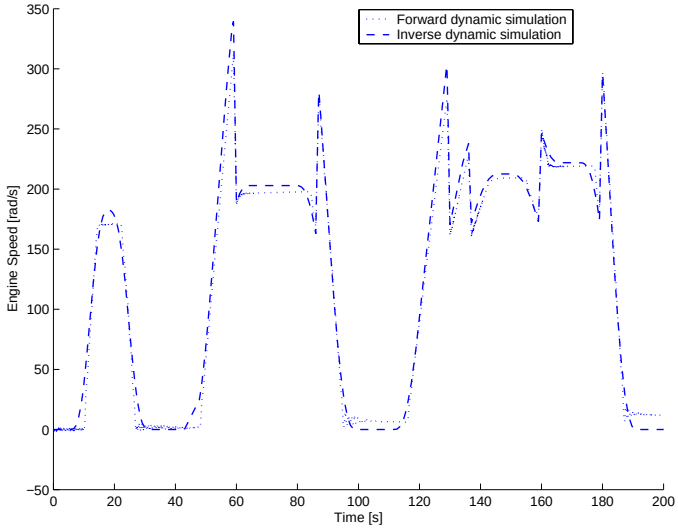

Figure 7.9 Engine speed of a simulation of the powertrain model with flexible drive shaft from Section 6.2 during a part of NEDC.

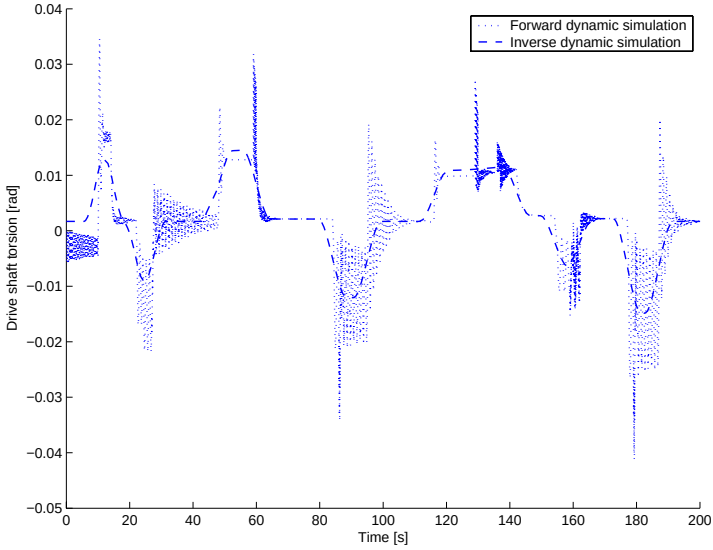

Figure 7.10 Drive shaft torsion of a simulation of the powertrain model with flexible drive shaft from Section 6.2 during a part of NEDC.

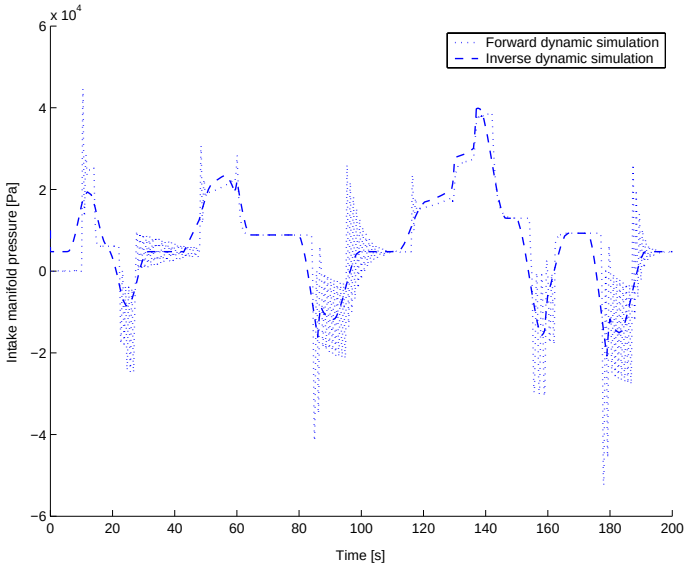

Figure 7.11 Intake manifold pressure of a simulation of the powertrain model with flexible drive shaft from Section 6.2 during a part of NEDC.

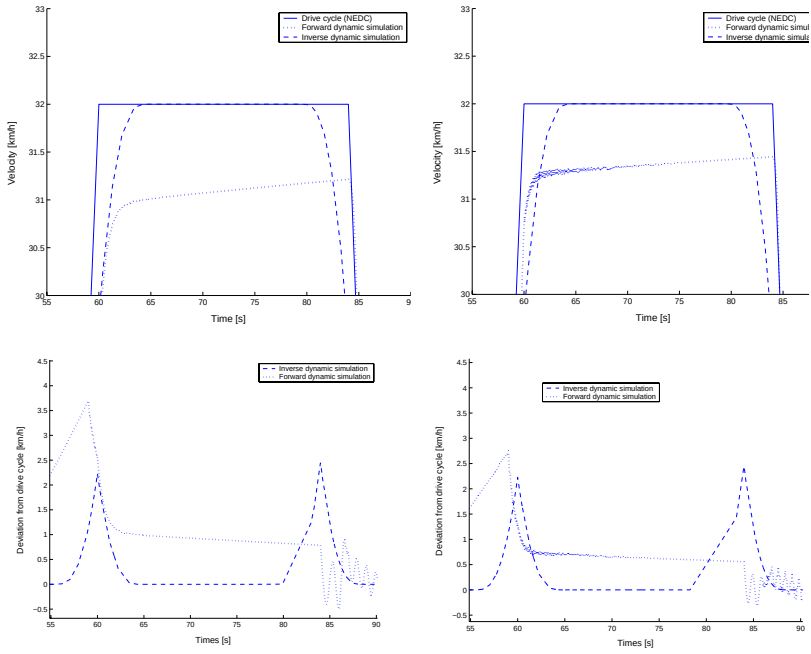

Figure 7.12 Velocities of simulations of two different vehicles in NEDC with the same driver models. To the left a 1700 kg vehicle with a 2.3 liter engine is simulated. To the right a 1100 kg vehicle with a 1.6 liter engine is simulated. The upper plots are the speed profiles and the lower plots are deviations from the drive cycle. Note that the deviation is the same in both cases for the inverse dynamic simulation, but differs for the two forward dynamic simulations.

case. This is a very unwanted behavior and shows that it can be difficult to run a set of forward dynamic simulations and get a good result without retuning the driver model when the vehicle parameters are changed. Of course these effects deteriorate the usefulness for the set of simulations as a design space exploration. However, for inverse dynamic simulation, due to the computational scheme with the implicit driver model the behavior within tracking limits are more consistent. The conclusion is that inverse dynamic simulation has a clear advantage regarding design space exploration.

7.5 Feasibility for non-minimum phase systems

So far two models have been simulated with both forward and inverse dynamic simulation, where the first model from Section 6.1 had no zero dynamics and the second model from Section 6.2 had stable zero dynamics. Both these models satisfied the requirements for invertibility.

Now the model from Section 6.3 with unstable zero dynamics is studied. The forward model is given by (6.45) and (6.46), and the inverse model is given by (6.49) and (6.50). The system is simulated around the stationary point $p_{im} = 2.06 \cdot 10^5\,\text{Pa}$, $p_{em} = 2.41 \cdot 10^5\,\text{Pa}$, $\omega_t = 7001\,\text{rad/s}$, $A_{egr} = 1 \cdot 10^{-4}\,\text{m}^2$, $u_{egr} = 1 \cdot 10^{-4}\,\text{m}^2$, $A_{vgt} = 4 \cdot 10^{-5}\,\text{m}^2$, $\lambda = 2.77$, and $x_{egr} = 7.1\%$. In the inverse simulation a coordinate change is done such that this point becomes the new origin. To compare the forward and inverse dynamic simulations, desired output trajectories, λ_d and $x_{egr,\,d}$, are specified as seen in Figure 7.13.

7.5.1 Simulations

Both forward and inverse simulation will be performed for this example in the following way: First the zero dynamics is simulated by solving (6.49), where five iterations are computed. The result is shown in Figure 7.14 where $\tilde{\eta}_1$ is to the left and $\tilde{\eta}_2$ is to the right. It is seen that the five iterations are almost on top of each other, which means that the method converges quickly for this problem. The inverse dynamic model (6.50) is then simulated which results in desired inputs $u_{egr,\,d}$ and $A_{vgt,\,d}$ as seen in

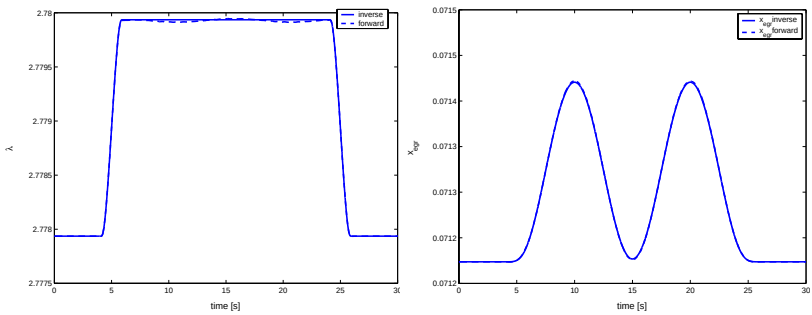

Figure 7.13 Specified output trajectories for λ and x_{egr} for the inverse dynamic simulation (solid) of the diesel engine model. Output from a corresponding forward dynamic simulation (dashed) is also seen. Note that the outputs deviate little from the stationary point.

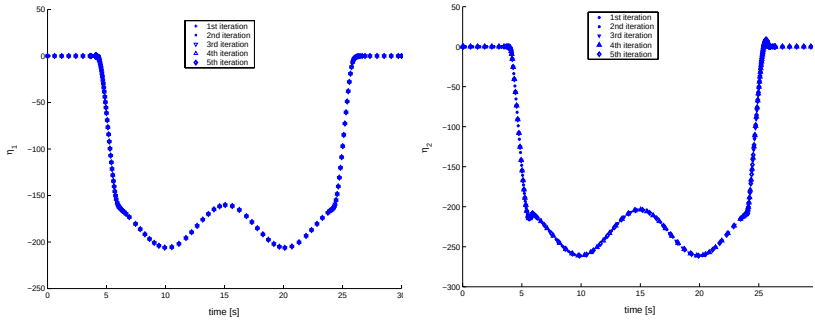

Figure 7.14 Five iterations of the separated zero dynamics $\tilde{\eta}$, (6.49), for the inverse simulation with desired output according to Figure 7.13. It is seen that after only a couple of iterations the trajectories are almost indistinguishable.

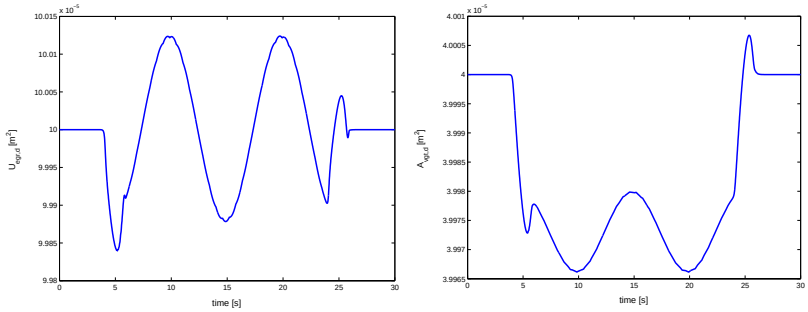

Figure 7.15 Required inputs from an inverse dynamic simulation of the diesel engine with desired outputs according to Figure 7.13. Note that the inputs deviates very little from the stationary point. These inputs are used in a forward dynamic simulation presented in Figures 7.13 and 7.16.

Figure 7.15. Using the inverse coordinate change the original states are also computed as seen in Figure 7.16.

Forward dynamic simulation is not easy since it is non trivial to design a good controller. Instead, the inputs resulting from inverse simulation in Figure 7.15, are used as inputs to the forward dynamic model. The results are shown in Figures 7.13 and 7.16. It is seen that the inverse dynamic simulation and the forward dynamic simulation coincide for this simulation.

As is mentioned in Section 3.1.2, for systems with highly nonlinear zero dynamics it can only be expected that the inverse dynamic simulation is correct for small deviations from the stationary point, the origin, and in the simulation above the deviation from the stationary point was small. How far from the origin is the inverse dynamic simulation correct? As an answer to this question the procedure described above, with an inverse dynamic simulation followed by a corresponding forward dynamic simulation, is carried

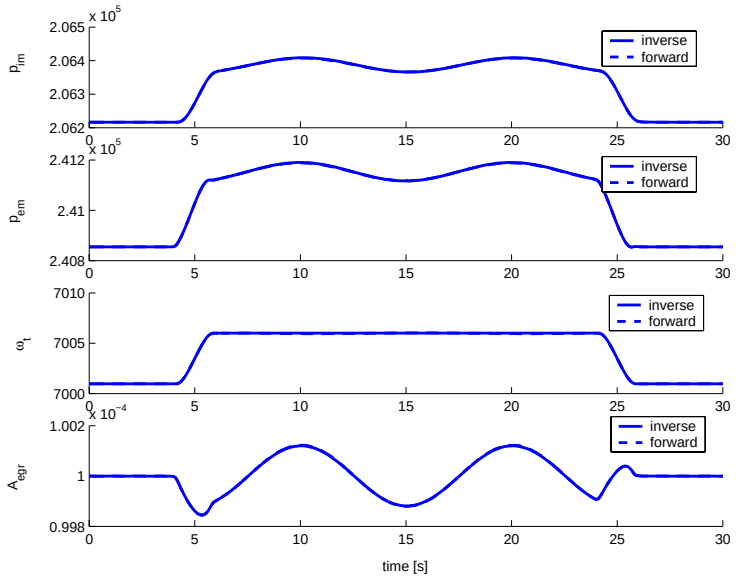

Figure 7.16 States from inverse and forward dynamic simulation of the diesel engine with inputs and outputs according to Figures 7.13 and 7.15. It is seen that the state trajectories coincide for both simulations. Note that the states deviate very little from the stationary point.

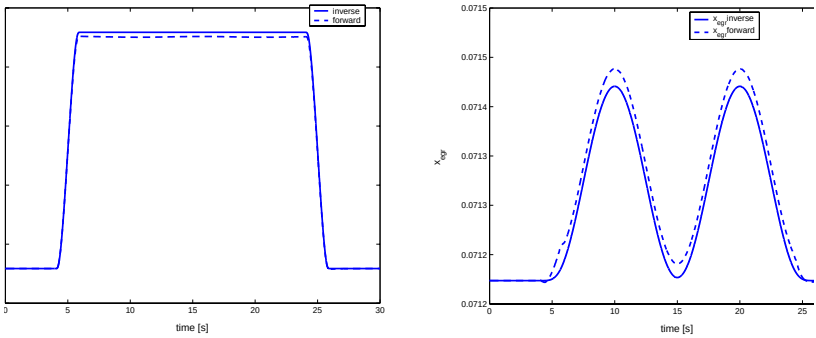

Figure 7.17 Specified output trajectories for the inverse dynamic simulation (solid) of the diesel engine model. Output from a corresponding forward dynamic simulation (dashed) is also seen. Note that the deviation in λ_d from the stationary point is ten times the deviation in Figure 7.13 and that the inverse and forward dynamic simulation does not produce the same result.

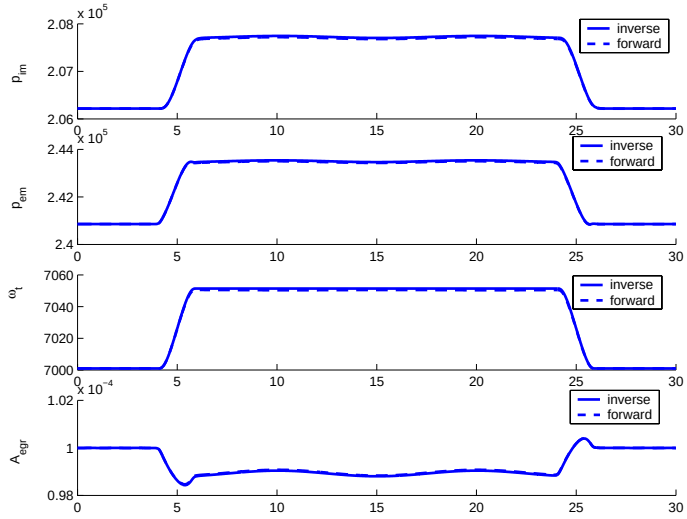

Figure 7.18 States from inverse and forward dynamic simulation of the diesel engine with outputs according to Figure 7.17. It is seen that the simulations does not give coinciding results

out with a larger deviation in λ_d, see Figure 7.17. The states from the simulations are shown in Figure 7.18. It is seen, mostly in the outputs λ and x_{egr}, that the results from the two simulations does not coincide. This is due to that the states and inputs computed from the inverse dynamic simulation does not satisfy the differential equation (6.45) which in turn is due to the fact that the linear approximation of the zerodynamics in (6.49) is not good enough in areas too far away from the origin.

To demonstrate that the method works well for linear systems a linearized version of the diesel engine model is used. A linearization of (6.45) and (6.46) results in

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & 0 & a_{24} \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & b_{22} \\ 0 & b_{32} \\ b_{41} & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Of course the eigenvalues of the zero dynamics is the same for this linear model and the nonlinear model in the linearization point. Since the zero dynamics now is linear and can be diagonalized, there is no need for an iteration to find the trajectories of it since the dynamics corresponding to the stable eigenvalue can be found directly by a causal simulation and the dynamics corresponding to the unstable eigenvalue can be found directly by a non-causal simulation. See Figures 7.19 and 7.20 for the simulation results of the linear model. There it is seen that even though the desired output trajectories are driven farther away from the linearization point, the simulation of the inverse and forward dynamic models coincide. However, since the linearization only describe the physics well in a neighborhood of the linearization point, the values of the states become unrealistic the further the desired outputs are from that point.

To conclude: For linear systems with unstable zero dynamics the method is easily applicable, but for non linear non minimum phase systems care has to be taken to make sure wether or not the results are accurate.

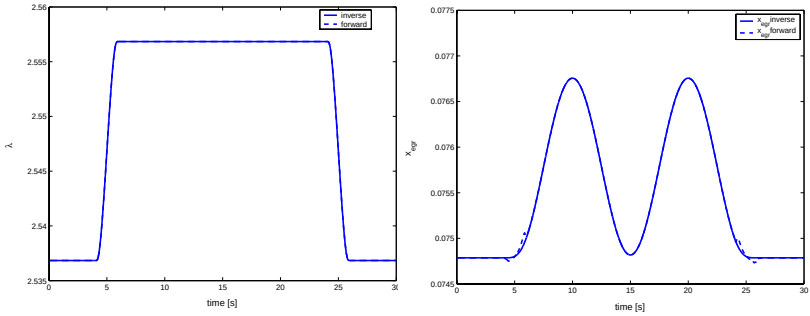

Figure 7.19 Specified output trajectories for the inverse dynamic simulation (solid) of the linearized diesel engine model. Output from a corresponding forward dynamic simulation (dashed) is also seen.

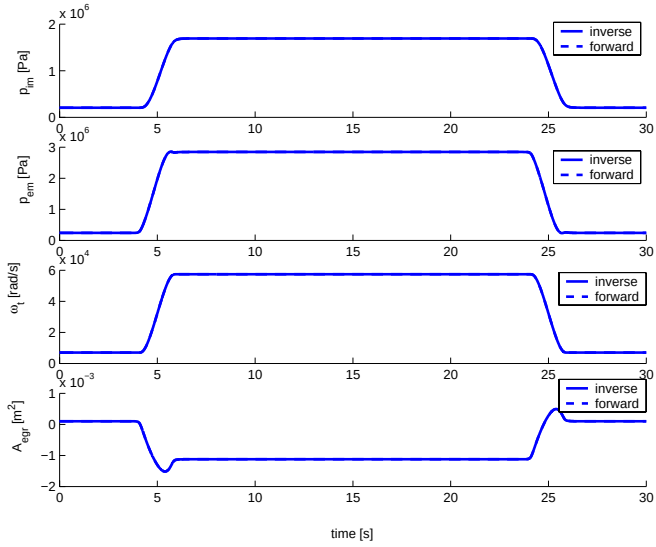

Figure 7.20 States from inverse and forward dynamic simulation of the diesel engine with outputs according to Figure 7.19. It is seen that the simulations does give coinciding results. Since the linear model only describe the physics good in a neighborhood of the linearization point, the values of the states are not physical when the outputs are driven away from that point.

7.6 Simulation time

Recall that the main goal was to extend quasi-static inverse simulation with additional dynamics while still keeping the low simulation time.

In order to study the simulation speed of inverse and forward dynamic simulation the three models from Chapter 6 will be used again in a more elaborate study including more cases than in the previous sections. Besides the drive cycle NEDC also FTP 75 has been used. Further, different driver models have been used as summarized in Tables 7.2 and 7.3. Twentysix combinations of vehicle models, driver models, and drive cycles have been simulated and they are listed in the presentation of the results in Table 7.1.

7.6.1 Approach in comparisons

The simulations presented in the results have been setup in the following way. The inverse dynamic simulations are not optimized for efficient simulation, but rather, in order to resemble an automatic conversion from the corresponding forward model, they are constructed using only the conversion described in Chapter 3. This means that there are additional potential for speed up, like to see if the inverse coordinate change can be solved analytically or if at least some of the coordinates can be solved analytically. Different choices of coordinates can also take different times to solve numerically.

For both inverse and forward simulation the solver has for each case been chosen to be the fastest available, and the resulting choice is listed in a column of the result presentation in Table 7.1

7.6.2 Timing results

The main result that can be seen in Table 7.1 is that the inverse dynamic simulation has a considerable speed advantage compared to the forward dynamic simulation. This holds for all three powertrain applications that have different added dynamics.

Going a bit more into detail there are a number of interesting observations to be made in the comparisons. There are mainly two things that affect the simulation time, the number of steps required in the integration, and the number of calculations in each step. In Table 7.1 it is seen that the inverse dynamic simulation takes at least an order of magnitude fewer steps than the forward dynamic simulation.

One reason that influences the difference in number of integration steps relates to charachteristics of the vehicle model. The number of steps in the integration is mainly depending on the eigenvalues of the simulated system, where large eigenvalues, i.e., far away from the imaginary axis, will require small steps and thus more steps. This means that when the eigenvalues of the zero dynamics is smaller than the eigenvalues for the forward dynamic model, the inverse dynamic simulation will require fewer steps than the forward and for many systems this is the case. For example, in the diesel

Model	Drivecycle	Simulation time [s]	# steps	Solver	Relative Tolerance	Forward/ Inverse
Basic powtrn 1	NEDC	31.4	9033	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 1	FTP75	51.5	14579	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 2	NEDC	22.5	6682	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 2	FTP75	37.0	10418	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 3	NEDC	18.2	5512	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 3	FTP75	31.6	9099	ode45	$1{\cdot}10^{-3}$	Forward
Basic powtrn 4	NEDC	0.31	50	ode45	$1{\cdot}10^{-3}$	Inverse
Basic powtrn 4	FTP75	0.071	50	ode45	$1{\cdot}10^{-3}$	Inverse
Basic powtrn 5	NEDC	0.064	50	ode45	$1{\cdot}10^{-3}$	Inverse
Basic powtrn 5	FTP75	0.070	50	ode45	$1{\cdot}10^{-3}$	Inverse
Basic powtrn 6	NEDC	0.033	50	ode45	$1{\cdot}10^{-3}$	Inverse
Basic powtrn 6	FTP75	0.069	50	ode45	$1{\cdot}10^{-3}$	Inverse
Flex powtrn 1	NEDC	551	344448	ode23tb	$1{\cdot}10^{-6}$	Forward
Flex powtrn 1	FTP75	302	130755	ode45	$1{\cdot}10^{-6}$	Forward
Flex powtrn 2	NEDC	429	222700	ode23tb	$1{\cdot}10^{-6}$	Forward
Flex powtrn 2	FTP75	292	125919	ode45	$1{\cdot}10^{-6}$	Forward
Flex powtrn 3	NEDC	365	186025	ode23tb	$1{\cdot}10^{-6}$	Forward
Flex powtrn 3	FTP75	277	120343	ode45	$1{\cdot}10^{-6}$	Forward
Flex powtrn 4	NEDC	80.9	10206	ode23tb	$1{\cdot}10^{-6}$	Inverse
Flex powtrn 4	FTP75	215	26762	ode23tb	$1{\cdot}10^{-6}$	Inverse
Flex powtrn 5	NEDC	40.8	5163	ode23tb	$1{\cdot}10^{-6}$	Inverse
Flex powtrn 5	FTP75	137	16805	ode23tb	$1{\cdot}10^{-6}$	Inverse
Flex powtrn 6	NEDC	156	16293	ode23tb	$1{\cdot}10^{-6}$	Inverse
Flex powtrn 6	FTP75	90.5	10848	ode23tb	$1{\cdot}10^{-6}$	Inverse
Diesel engine	-	12.9	3596	ode23tb	$1{\cdot}10^{-6}$	Forward
Diesel engine	-	8.66	2201	ode23tb	$1{\cdot}10^{-6}$	Inverse

Table 7.1: Simulation times. The “Basic powtrn” models is the model of a basic powertrain without zero dynamics from Section 6.1. The “Flex powtrn” is the model of a powertrain with a flexibility in the driveshaft from Section 6.2. These models use driver parameters according to Tables 7.2 and 7.3. The “Diesel engine” model is the one with unstable zero dynamics described in Section 6.3. One iteration has been used to find the zerodynamics of the inverse diesel engine model. The number of integration steps in the inverse simulation of the diesel engine is the sum of steps for the simulations of the zero dynamics and for finding the required inputs.

Model	K_p	K_i	K_d
Basic powtrn 1	$6 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	0
Basic powtrn 2	$1.5 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	0
Basic powtrn 3	$8 \cdot 10^{-5}$	$1 \cdot 10^{-6}$	0
Flex powtrn 1	$2 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	0
Flex powtrn 2	$1.5 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	0
Flex powtrn 3	$8 \cdot 10^{-5}$	$1 \cdot 10^{-6}$	0

Table 7.2: Driver parameters for the forward dynamic models of Table 7.1.

Model	Driver parameter *a*
Basic powtrn 4	1
Basic powtrn 5	2
Basic powtrn 6	5
Flex powtrn 4	1
Flex powtrn 5	2
Flex powtrn 6	5

Table 7.3: Driver parameters for the inverse dynamic models of Table 7.1.

engine model the eigenvalues of the forward model in the linearization point are -18.2, -12.7, -5.00, -1.26, and for the zero dynamics the eigenvalues are 15.8 and -6.8. For the basic powertrain model there is no zero dynamics, so the inverse model is correct no matter how few steps that are taken. For the flexible powertrain model the zero dynamics stems from the drive shaft flexibility that has slower dynamics than e.g. the intake pressure dynamics.

Another reason that influence the difference in simulation time is the calculations needed in each integration step. Since the dimension of the zero dynamics is smaller than the number of states in the forward dynamic models the number of calculations in each step are fewer in most cases. If however there is a need to solve the inverse coordinate change numerically online, the number of calculations in each step can increase some for the inverse dynamic models.

The main reason for the relatively slow simulations of the forward models of the basic powertrain and the flexible powertrain is the need for extra dynamics in the form of a controller, the driver model. Typically, such a controller has ten times faster dynamics than the controlled system. Typical plots from simulations presented in Table 7.1 have been presented in previous sections and are seen in Figures 7.1 - 7.20. From those plots it might seem that the driver model in the forward dynamic simulation is unnecessarily oscillating in some cases, but the trends in simulation time reported in Table 7.1 is still valid.

7.7 How to choose simulation method

As seen in the previous sections there are many aspects to consider when choosing simulation method. Based on the simulation studies made during this work some, sometimes subjective, experiences will now be given. First to consider when a model is to be simulated is why the simulation is done. Other issues are, what result that is wanted and to what cost, what accuracy that is needed, how long time that is available, how many simulations of the same model that is considered, and how much time that can be spent on setting up the simulation vs. how long time that can be spent on the simulation.

The accuracy of inverse and forward dynamic simulation respectively has to be considered. The theory about accuracy of forward dynamic models, or ODE's, is well known, see e.g. [2, 16]. Inverse dynamic models are in most cases also ODE's, so the same theory can be applied. The exceptions are systems without zero dynamics where the inverse model becomes a set of algebraic equations. Such equations are easier to solve to a higher precision using numerical methods such as Newton type methods, see [6].

What is an appropriate solver? Of course nothing can be said in general since the choice of solver is model dependant. A good guide for choosing solvers is [2]. However, from the examples studied, some conclusions can be drawn. For simple models like the basic powertrain an explicit Runge-Kutta solver like matlab's ODE45 is a good choice to try. When the inverse coordinate change is solved numerically online an extra error is added in each step. This can be seen as noise or very fast dynamics which makes the system behave as a stiff system, and hence an implicit solver with stiff decay is suitable.

As mentioned earlier it is important to consider drive cycle tracking. For the forward dynamic simulation it is more work in desiging a driver model than in inverse dynamic simulation. In fact, designing a controller for a forward dynamic simulation has about the same complexity as setting up the inverse simulation. Since one has perfect knowledge of the system that is going to be controlled in a simulation, one natural choice of a controller for the forward simulation is to use the system inverse as a feed forward term. This feed forward term however, is the inverse dynamic model.

Conditions, like e.g. if-statements, are more straight forward to handle in forward dynamic models, but, as has been shown in the examples, are often straightforward to handle in inverse dynamic models as well. For nonlinear non minimum phase systems, care has to be taken so that the simulation does not drive the zero dynamics out of the region from where the iteration converges to the right solution.

The conclusions of the comparison of inverse- and forward dynamic simulation is summarized in Table 7.4. There it is seen that inverse dynamic simulation is faster and more suited for batch simulations than forward dynamic simulation. Since it is hard to define a driver model in the forward dynamic

simulation it is in many cases easier to set up an inverse dynamic simulation. However, forward dynamic simulation is capable of simulating a larger class of systems than inverse dynamic simulation, e.g., highly non linear non-minimum-phase systems.

Method	Simulation speed	Simulation batch applicability	Simulation set up	Size of applicable model class
Inverse dynamic simulation	+	+	+	-
Forward dynamic simulation	-	-	-	+

Table 7.4: Comparison of inverse- and forward dynamic simulation. A plus sign indicates that the method has good properties in that area, and a minus sign that the method has less good properties.

8

Conclusions

As mentioned before, drive cycle simulation of vehicle models is a commonly used tool for driveline design, driveline optimization, and, design of driveline control strategies. When studying for example drivability, fuel consumption, or engine emissions, it is important that the models describe the dynamics of the powertrain to a high accuracy. To perform fast simulations of such powertrain models inverse dynamic simulation was proposed. In this method the system has to be inverted, and inversion of non linear systems was discussed in Chapter 3.

Drive cycle tracking and driver models were discussed in Chapter 4. There a new inverse driver model was presented that together with the inversion procedure from Chapter 3 forms the new concept of inverse dynamic simulation. The use of the new driver model is a good way of specifying tracking behavior. Compared to specifying tracking using the forward driver model it is closer to human behavior when performing drive cycle tests on a chassis dynamometer, since it is not specified by controller parameters, but rather in terms of tracking smoothness.

Regarding inversion, by use of symbolic math tools, the inversion procedure can be more or less automatized. The systematic conversion reduces the need to build dedicated inverse models as those used in, e.g., Advisor. In this way the new simulation concept is a bridge that ties forward dynamic simulations as in, e.g., PSAT, to quasi-static inverse simulations as in, e.g., Advisor, in the sense that common model data bases can be used.

There is an ambiguity in simulation results due to the fact that drive cycle tracking requirements are specified within limits. Therefore, performance measures, needed to compare inverse dynamic simulation with forward dynamic simulation, were discussed in Chapter 5.

In Chapter 6 three powertrain applications was presented that included important dynamics that can not be handled using quasi-static inverse simulation. The extensions were engine dynamics, drive line dynamics, and gas flow dynamics around diesel engines. These three cases also represented interesting mathematical properties such as zero dynamics, resonances, and non-minimum phase systems, i.e. unstable zero dynamics. The inversion technique was demonstrated on all three examples, and in Chapter 7 the feasibility of inverse dynamic simulation of these systems was shown. Moreover, in Chapter 7, using the three examples, inverse dynamic simulation was compared to forward dynamic simulation regarding simulation set-up effort, simulation time, and parameter-result dependency. It was shown that inverse dynamic simulation was easy to set up, gave short simulation times, and gave consistent result for design space exploration. This makes inverse dynamic simulation a suitable method to use for drive cycle simulation, and especially in situations requiring many simulations, such as optimization over design space, powertrain configuration optimization, or development of powertrain control strategies.

9

Nomenclature

A	Vehicle cross sectional area
A_{eff}	Effective throttle area
A_{egr}	EGR valve area
A_{th}	Throttle area
A_{vgt}	VGT area
$\left(\frac{A}{F}\right)_s$	Stoichiometric air fuel ratio
B	Engine bore
$BMEP$	Brake mean effective pressure
C	Discharge coefficient
c_d	Air drag coefficient
c_{ds}	Drive shaft damping
c_{pa}	Specific heat under constant pressure, ambient conditions
c_{pe}	Specific heat under constant pressure, exhaust gases
c_r	Rolling resistance coefficient
F	Wheel traction force
$FMEP$	Friction mean effective pressure

g	Gravitational acceleration
$IMEP_g$	Gross indicated mean effective pressure
i	Gear ratio
J_e	Engine inertia
J_t	Turbo inertia
k_{ds}	Drive shaft stiffnes
M_c	Compressor torque
M_t	Turbine torque
m	Vehicle mass
m_{ac}	Engine air charge
$\dot{m}_{at}$	Air mass flow past throttle
m_f	Engine fuel charge
N	Engine speed
n_r	Revolutions per engine cycle
p_a	Ambient pressure
p_e	Exhaust manifold pressure
p_i	Intake manifold pressure
p_r	Pressure ratio
$PMEP$	Pump mean effective pressure
q_{hv}	Fuel heating value
r	Wheel radius
R	Gas constant
R_a	Gas constant ambient conditions
R_c	Compressor radius
R_e	Gas constant exhaust manifold
s_i	Slope of the normalized air charge
S	Engine stroke
T	Torque, Temperature
T_a	Ambient Temperature
T_{cr}	Crank shaft torque
T_d	Drive shaft torque
T_e	Engine (flywheel) torque
T_{em}	Exhaust manifold temperature
T_i	Intake manifold temperature
T_w	Wheel torque
u_{egr}	Control signal to EGR throttle

u_δ	Injected mass of fuel
v	Vehicle speed
V_d	Engine displacement volume
V_e	Exhaust manifold volume
V_i	Intake manifold volume
W_c	Mass flow through compressor
W_{egr}	Mass flow EGR
W_{ei}	Mass flow into the engine
W_{eo}	Mass flow out of the engine
W_f	Fuel mass flow
W_t	Mass flow through the turbine
y_i	y-intercept of the normalized air charge
γ	Specific heat ratio
η_{th}	Thermal efficiency
η_{tm}	Transmission efficiency, combined mechanical and turbine efficiency
η_{vol}	Volumetric efficiency
θ_d	Drive shaft torsion
λ	Normalized air fuel ratio
ΔT_c	Temperature difference over compressor
ΔT_t	Temperature difference over turbine
ξ_{aux}	Friction coefficient
π_{bl}	Boost layout
ρ	Air density
τ_{egr}	Time constant for EGR throttle
Φ_c	Volumetric flow coefficient, compressor
ω	Rotational Speed
ω_e	Engine rotational speed
ω_t	Turbo speed
ω_w	Wheel rotational speed

Parameter values for the powertrain models, SI-units

A	2.5
$\left(\frac{A}{F}\right)_s$	14.8
B	0.09
c_d	0.33
c_{ds}	3.183
c_r	0.01
g	9.81
i	[13.1482, 6.8464, 4.5902, 3.4621, 2.7320]
J_e	0.2
k_{ds}	29794
m	1700
n_r	2
p_a	$1 \cdot 10^5$
p_e	$1 \cdot 10^5$
q_{hv}	$44.651 \cdot 10^6$
r	0.3
R	286
s_i	0.95
S	0.09
T_a	293
T_i	293
V_d	$2.3 \cdot 10^{-3}$
V_i	$2 \cdot 10^{-3}$
y_i	$-0.09 \cdot 10^5$
γ	1.26
η_{th}	0.3604
η_{tm}	0.9
ξ_{aux}	1.34
π_{bl}	1
ρ	1.29

10

References

[1] J. Andersson. *Multiobjective optimization in engineering design.* PhD thesis, Linköping Institute of Technology, thesis No. 675, ISBN 91-7219-943-1. 2001.

[2] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations.* SIAM. 1998.

[3] M. Back, S. Terwen and V. Krebs. *Predictive powertrain control for hybrid electric vehicles.* 1st IFAC symposium on advances in automotive control, Salerno, 2004

[4] J. Bals, G. Hofer, A. Pfeiffer, and C. Schallert. *Object-Oriented Inverse Modelling of Multi-Domain Aircraft Equipment Systems and Assessment with Modelica.* 3rd International Modelica Conference, Linköping, Sweden, November 3-4. 2003

[5] K. L. Butler, M. Ehsani and P. Kamath. *A Matlab-Based Modeling and Simulation Package for Electric and Hybrid Electric Vehicle Design.* IEEE Transactions on Vehicular Technology, vol 48, 1770-1778. 1999.

[6] G. Dahlquist and Å. Björk. *Numerical Mathematics and Scientific Computation.* http://www.mai.liu.se/~ akbjo/NMbook.html (to be published by SIAM, Philadelphia) 2005.02.03

[7] S. Devasia, D. Chen and B. Paden. *Non-linear inversion-based output tracking.* IEEE Transactions on Automatic Control vol 41, 930-942, 1996

[8] S. Devasia and B. Paden. *Stable Inversion for Nonlinear Nonminimum-Phase Time-Varying Systems.* IEEE Transactions on Automatic Control vol 43, 283-288. 1998.

[9] S. Devasia, B. Paden and C. Rossi. *Exact-output Tracking Theory for Systems with Parameter Jumps.* International Journal of Control, vol 67, 117-131. 1997.

[10] L. Eriksson. *Simulation of a Vehicle in Longitudinal Motion with Clutch Lock and Clutch Release.* 3rd IFAC workshop on advances in automotive control. 65-70. 2001.

[11] A. Fröberg and L. Nielsen. *A Method to Extend Inverse Dynamic Simulation of Powertrains with Additional Dynamics.* First IFAC Symposium on Advances in Automotive Control. 2004.

[12] A. Fröberg and L. Nielsen. *Dynamic Vehicle Simulation -Forward, Inverse and New Mixed Possibilities for Optimized Design and Control.* SAE Technical Paper Series SP-1826, 2004-01-1619. 2004.

[13] A. Fröberg, L. Nielsen, L-G Hedström, and M. Pettersson. *Controlling gear engagement and disengagement on heavy trucks for minimization of fuel consumption.* 16th IFAC world congress, Prague. 2005.

[14] K. Gustafsson. *Traps and Pitfalls in Simulation.* Ericsson Mobile Communications AB, Lund, Sweden.

[15] L. Guzzella and A. Amstutz. *CAE Tools for Quasi-Static Modelling and Optimization of Hybrid Powertrains.* IEEE Transactions on Vehicular Technology, vol 48, 1762-1769. 1999.

[16] E. Hairer, S.P. Norsett and G. Wanner. *Solving Ordinary Differential Equations 1, nonstiff problems.* Springer. 1993.

[17] E. Hendricks, A. Chevalier, M. Jensen, S. C. Sorenson, D. Trumpy and J. Asik. *Modelling of the Intake Manifold Filling Dynamics.* SAE Technical Paper Series Sp-1149, 1-25. 1996.

[18] J. B. Heywood. *Internal Combustion Engine Fundamentals.* McGraw/Hill. 1988.

[19] L. R. Hunt and G. Meyer. *Stable Inversion of Nonlinear Systems.* Automatica, vol 33, 1549-1554. 1997.

[20] A. Isidori. *Nonlinear Control Systems.* Springer/Verlag 1995.

[21] U. Kiencke and L. Nielsen. *Automotive Control Systems.* Springer Verlag. 2000.

[22] D.J. Murray-Smith. *The inverse simulation approach: a focused review of methods and applications.* Mathematics and computers in simulation. 2000.

[23] M. Pettersson and L. Nielsen. *Gearshifting by engine control.* IEEE Transactions Control Systems Technology, Volume 8(No. 3) pp. 495-507, 2000.

[24] M. Pettersson and L. Nielsen. *Diesel engine speed control with handling of driveline resonances.* Control Engineering Practice, 11(No. 10) pp. 319-328, 2003.

[25] G. Rodriguez. *Kalman Filtering, Smoothing, and Recursive Robot Arm Forward and Inverse Dynamics.* IEEE Journal of Robotics and Automation, vol RA-3, no 6. 1987.

[26] A. Rousseau, P. Sharer and F. Besnier. *Feasability of Reusable Vehicle Modeling: Application to Hybrid Vehicles.* SAE Technical Paper Series SP-1826. 2004-01-1618 2004.

[27] T. Sandberg. *Heavy truck modeling for fuel consumption. Simulation and measurements.* Licentiate thesis. Linköping Institute of Technology. Thesis No. 924. LiU-TEK-LIC-2001:61. 2001.

[28] A. Sciarretta, L. Guzzella, and C. H. Onder. *On the power split control of parallel hybrid vehicles: from global optimization towards real-time control.* Automatisierungstechnik 51 (5), 2003.

[29] P. Soltic. *Part/Load Optimized SI Engine Systems.* PhD thesis. Swiss Federal Institute of Technology. 2000.

[30] D. G. Thomson and R. Bradley. *The principles and practical application of helicopter inverse simulation.* Simulation Practice and Theory vol 6, no 1. 1998.

[31] J. Wahlström. *Modeling of a diesel engine with VGT and EGR.* Technical Report, Vehicular Systems, Department of Electrical Engineering, Linköping University, Sweden. 2004.

[32] C. W. Wampler. *Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods.* IEEE Transactions on systems, man, and cybernetics, vol. smc-16 no 1. 1986.

[33] K. B. Wipke, M. R. cuddy and S. D. Burch. *ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach.* IEEE Transactions on Vehicular Technology, vol 48, 1751-1761. 1999.

[34] G. D. Wood. *Simulating Mechanical Systems in Simulink with SimMechanics.* www.mathworks.com.

[35] *www.capsim.se* 2005.02.08

Notes: