An embedded model predictive controller for optimal truck driving

FRANCESCO MANCINO
Abstract

An embedded model predictive controller for velocity control of trucks is developed and tested.
By using a simple model of a heavy duty vehicle and knowledge about the slope of the road ahead, the fuel consumption while traveling near a set speed is diminished by almost 1% on an example road compared to a rule based speed control system.
The problem is formulated as a look-ahead optimization problem were fuel consumption and total trip time have to be minimized. To find the optimal solution dynamic programming is used, and the whole code is designed to run on a Scania gearbox ECU in parallel with all the current software.
Simulations were executed in a Simulink® environment, and two test rides were performed on the E4 motorway.
Sammanfattning

Acknowledgments

I would like to express my sincere gratitude towards my two supervisors at Scania: Björn Johansson and Henrik Svärd. Without their knowledge, engagement, and positivity this project would not have been possible. I also want to thank all the other members of the NEPP group for warmly welcoming me at Scania and making me feel a part of the team.

Special thanks go to Oscar Flärdh and Martin Jakobsson at NECS department at Scania for their substantial help during the development of the controller; it has been of great value to be able to collaborate in such an open way between departments.

I would like to thank my supervisor at KTH, Arda Ayetekin, for his support and availability to answer any questions I had. I also want to thank Professor Michael Johansson for starting this project and believing in me.

I would also like to thank Xiao Cheng, who worked with a parallel project and has been a great partner to share information and tribulations.

Last but not least I want to thank my family for believing in me all these years and always supporting me!

Stockholm, October 2016

Francesco Mancino
Contents

1 Introduction 4
  1.1 Background .......................... 5
  1.2 Structure of the report ................. 6

2 Truck model 7
  2.1 Drive force .......................... 7
  2.2 Resistance .......................... 9
  2.3 Dynamics of the truck ................. 10

3 Optimization problem statement 11
  3.1 Propulsive work model .................. 11
    3.1.1 Calculation of the penalty parameter $\beta$  12
  3.2 Fuel model .......................... 13

4 Dynamic programming solution 15
  4.1 Discretization ........................ 15
  4.2 Receding horizon ...................... 16
  4.3 Dynamic programming Algorithm ........ 17
    4.3.1 Constant gear modeling .......... 17
    4.3.2 Gear shift modeling ............. 19
    4.3.3 Calculation of the trajectory .... 20
    4.3.4 Handling differences in torque .. 21
    4.3.5 Handling interpolation at boundaries ... 23
  4.4 Memory requirements .................... 24

5 Simulation Results 26
  5.1 Simulink® simulation environment .... 26
  5.2 Simulations .......................... 26
    5.2.1 Analysis of the driving strategy . 27
    5.2.2 Energy analysis .................. 32
    5.2.3 Issues .......................... 33

6 Trial run 36
  6.1 Test setup .......................... 36
  6.2 Results ............................ 37

7 Conclusions 39
  7.1 Future work .......................... 39
Nomenclature

\[ A_f \] Frontal area of the truck \([m^2]\)
\[ \beta \] Penalty parameter: trade-off between fuel consumption and trip time
\[ c \] Mass factor from rotational inertia inside the truck
\[ C_D \] Coefficient of aerodynamic resistance
\[ c_f \] Fuel consumption \([g/s]\)
\[ \text{DP} \] Dynamic Programming
\[ \text{Eco-roll} \] To drive in neutral gear
\[ \text{ECU} \] Electronic control unit
\[ e \] Kinetic energy of the truck \([J]\)
\[ F_a(v) \] Air drag \([N]\)
\[ F_d \] Total drag force \([N]\)
\[ F_c \] Forward driving force \([N]\)
\[ F_{err} \] Correction force for drag forces \([N]\)
\[ F_g(s) \] Gravitational force \([N]\)
\[ F_r \] Rolling resistance \([N]\)
\[ M_f \] Total fuel consumption \([g]\)
\[ f(T_e, v) \] Fuel consumption as a function of drivetorque and velocity \([g/s]\)
\[ g \] Gear
\[ \gamma \] Energy equivalent constant (fuel to energy)
\[ W_{\text{fuel}} \] Energy contained in the consumed fuel \([J]\)
\[ \delta g \] Gearshift
\[ G \] Set of allowed gears
\[ \text{GPS} \] Global Positioning System
\[ g_r \] Gravitational acceleration \([m/s^2]\)
HDV Heavy-duty vehicle

$i(g)$ Conversion ratio of the gearbox and final drive

$J_n$ Cost-to-go at stage $n$

$J_e$ Inertia in the engine [kg·m$^2$]

$J_t$ Lumped inertia of the gears and final drive [kg·m$^2$]

$m$ Mass of the truck [kg]

CCMP Model predictive cruise controller - Developed in this report

$\omega_e$ Engine speed [rpm]

CCRB Rule based look-ahead cruise controller - Used as a baseline

$r_w$ Radius of the wheels of the truck [m]

$s$ Position of the vehicle [m]

$T_0$ Maximum trip time [s]

$T_f$ Flywheel torque [Nm]

$T_{tot}$ Total torque generated by the engine (Flywheel+ friction) [Nm]

$v$ Velocity of the truck [m/s]

$W$ Propulsive work [J]

$\xi_n$ Step cost from stage $n$ to stage $n + 1$, or terminal cost if $n$ is the last stage.
Chapter 1

Introduction

Heavy-duty vehicles (HDVs) are of vital importance for economic growth. Almost half of all the transported goods in the world are carried over roads \cite{European-Commission:2015}, and this is not likely to change in the future since road transportation allows for a flexibility that is unmatched by other means of transport.

However, due to the burning of fossil fuel, HDVs are also responsible for a significant share of the global CO$_2$ gas emissions. Road transport in general stands for about 20\% of the total greenhouse gas emissions in Europe and HDVs stand for about one-fourth of the total CO$_2$ emissions by road vehicles \cite{European-Commission:2015}. Similarly, the fuel cost amounts to about 30\% of the total life cost of an HDV \cite{Schittler:2003}. Given an average of 130,000 km per year driven by a heavy duty vehicle \cite{Hill:2011}, an average fuel cost of 1.5 euro/liter and an efficiency of 3 km/l the fuel cost for a single vehicle amounts to 65000 euros a year \cite{Turr:2015}. It follows that diminishing the fuel consumption by a mere one percent would translate in approximately 650 euros saved per year per vehicle. Scania sold about 70 thousand HDVs in 2015, so it’s easy to see how small saving in fuel consumption could translate into a better environment and more profitable businesses.

This report investigates the possibility of implementing an optimization based cruise controller in the embedded software of a Scania truck.

A standard cruise controller aims to keep the desired speed over the whole drive mission, which is fuel efficient while driving on a flat road but has clear disadvantages when it comes to more hilly roads. Consider a road profile like the one in Figure \ref{fig:1} with a standard cruise controller set at 80 km/h. The truck would reach point $a$ at 80 km/h and, if the downhill segment in front is long and steep enough, it will need to brake during the descent to keep the speed constant. Later, arriving at point $c$ at 80 km/h the controller is probably going to apply full torque to get over the hill in front, and maybe a gearshift will be needed to keep the speed constant over the whole uphill segment.

A more natural and efficient way of driving would be to let the truck slow down a little before point $a$, let it roll to a higher speed in the downhill segment, and, assuming the distance between $b$ and $c$ is short, keep this speed until the start of the uphill section. Having a higher speed at the start of the uphill segment will help the truck get over the hill without gear-shifting and applying lower torque, but of course without keeping the speed constant. By doing this proce-
It is possible to maintain the same mean speed consuming far less fuel. Since every new Scania truck has GPS data about the road ahead, all the data are there for the vehicle to implement an efficient style of driving. The aim of this report is to formulate and solve an optimization problem where the fuel consumption is to be minimized, but the mean speed over the mission is kept constant. The formulation should be efficient enough to be implemented on a Scania embedded controller, but still give significant savings and provide an acceptable driving experience.

![Figure 1.1: A possible road profile the truck could experience during a drive mission. Clearly, keeping the same speed over this entire interval would not be optimal for fuel consumption.](image)

### 1.1 Background

The first formulation of an optimization problem to minimize fuel consumption using the road profile ahead of the vehicle was done in [Schwarzkopf and Leipnik](1977), where the authors analyzed specific road profiles and found a solution using Pontryagin’s minimum principle. This work was later revised by [Chang and Morlok](2005). Subsequently, [Fröberg et al.](2006) also provided analytic solutions for specific road profiles, using other methods. One of the main outcomes of the last two mentioned studies is that for constant slope the optimal velocity is constant, as long as linear relationship between fuel consumption and driving torque is assumed. In a later study, [Ivarsson et al.](2009) showed that considering non-linearities in specific fuel consumption could produce “forbidden” speeds, which are not optimal on specific road grades. The solution when either of these speeds is requested as a hard constraint would then give a combination of two speeds that result in a more efficient combustion.

Numerical methods have also been used to solve this problem, most notably dynamic programming (DP). The first paper to suggest this methodology was written by Hooker in 1988. In this study, both driving over hills and optimal acceleration problems were analyzed. The algorithm used was relatively slow, with three states (speed, position and gear), time as a stage, and an objective function based only on fuel economy. Mean speed was guaranteed by fixing the total trip time and the end position, and forward dynamic programming was used to solve the problem.

Significant improvements in computational efficiency were obtained by [Monastyrsky and Golownykh](1993). In this paper, the hard constraint on time is
relaxed by transferring the time component inside the objective function with a penalty parameter, \( \beta \). This operation consents to solve the problem without using time as a variable, having position as a stage, and only two states: speed and gear. While speeding up computations significantly, the possibility to fix the mean speed is lost. It can be done indirectly by sizing \( \beta \) appropriately.

In Hellström et al. (2006) a predictive cruise controller implementing an algorithm based on Monastyrsky and Golownykh formulation of the optimal driving problem is developed and subsequently tested in Hellström et al. (2009). Later, a new faster DP algorithm is developed in Hellström et al. (2010). This algorithm was tested on Scania trucks, and run very fast on a PC, but never got implemented into an embedded system. A couple of years later, in 2012, Scania trucks equipped their HDVs with a new Active Prediction cruise control system. This cruise control uses GPS data to calculate an efficient speed trajectory over major hills. It achieves up to 3% fuel savings compared to a traditional cruise controller, showing the great potential this technology has. Later, in 2014, Scania Active Prediction was equipped with the Eco-roll function, which allows the truck to coast in neutral gear down certain hills, allowing for further savings in fuel consumption.

With the evolution of embedded hardware inside the modern trucks, it is now interesting to investigate if it is possible to solve a mathematical optimization problem online using standard hardware, and compare the results with the systems in use right now. In this report, a solution of the problem based on DP is studied.

1.2 Structure of the report

The structure of the thesis is as follows:

In the next chapter, “Truck Model”, the modeling of the dynamics of the truck is described in depth, as well as how the torque output is related to fuel consumption.

In Chapter 3, “Optimization Problem statement”, the optimization problem is going to be defined mathematically, and an introduction on how to calculate the penalty parameter \( \beta \) is given.

In Chapter 4, “Dynamic programming solution”, the numerical solution Algorithm, based on Dynamic Programming (Bellman, 1957), is described.

In Chapter 5, “Simulation results”, the results from computer simulations on specific road profiles are presented.

In Chapter 6, “Field testing”, the experimental setup is described, and data from the real driving tests is presented.

In Chapter 7, “Conclusion”, the results are presented and analyzed, and suggestions about improvements and future work are included.
Chapter 2

Truck model

The truck modeling follows Sandberg (2001) and Hellström et al. (2010), and the resulting model is reformulated and adapted for the numerical optimization that is performed.

2.1 Drive force

The first component of the model is the flywheel torque, $T_e$. This is the torque that will be transmitted from the engine to the gearbox. The flywheel torque ($T_e$) is the result of the total produced torque ($T_{\text{tot}}$), which is directly dependent on fuel consumption ($c_f$), minus the friction in the engine ($T_f$), which is linearly dependent on the engine speed ($\omega_e$).

$$T_e(c_f, \omega_e) = T_{\text{tot}}(c_f) - T_f(\omega_e).$$

Clearly, $T_e$ has upper bounds and lower bounds depending on engine speed as can be seen in Figure 2.1.

The fuel consumption ($c_f$) can be looked up in a table as a function of ($T_e$) and ($\omega_e$), as illustrated in Figure 2.2.

This torque will then be transmitted through the gearbox and the final drive to the wheels, where it will generate a driving force ($F_e$).

$$F_e(T_e, T_b, \omega_e, g) = \frac{T_e i(g) - T_b}{r_w c},$$

(2.1)

where $g$ is the currently engaged gear, $i(g)$ is the conversion ratio of the gearbox and the final drive, $r_w$ is the wheel radius, $T_b$ is the breaking torque, and $c$ is a mass factor, that takes in consideration the rotation of components inside the truck:

$$c = 1 + \frac{J_l + i^2 J_e}{m r_w^2},$$

$J_l$ is the lumped inertia of the wheels and the final drive, and $J_e$ is the inertia of the motor.

Since $\omega_e$ is related to the rotational speed of the wheel ($\omega_w$), we can write:

$$\omega_e = \frac{\omega_w i}{r_w} = \frac{v i(g) 60}{r_w 2 \pi}.$$  

(2.2)
Figure 2.1: Maximal and minimal torque generated by the engine at different engine running speeds.

Figure 2.2: Fuel consumption \([\text{g/s}]\) as a function of engine speed and torque. Blue stands for low fuel consumption, whereas yellow for higher...
2.2 Resistance

This driving force will meet resistance from the air drag \( F_a(v) \), the rolling resistance \( F_r \), and the gravitational force \( F_g(s) \), as seen in Figure 2.3.

![Figure 2.3: A schematic image of the forces acting on a truck](image)

The air drag, is calculated using the following relation:

\[
F_a(v) = \frac{\rho}{2} C_D A_f v^2,
\]

where \( \rho \) is the mass density of the air, \( C_D \) is the coefficient of aerodynamic resistance (different for every object), \( A_f \) is the frontal area of the truck, and \( v \) is the velocity of the vehicle.

The roll drag is assumed to be constant:

\[
F_r = C_r m g_r.
\]

This formula is based on the ISO18164 standards for measuring the rolling resistance, \( C_r \) being the coefficient of rolling resistance in the standardized test. \( m g_r \) is the normal force from the truck. It was chosen not to take into account the differences in normal force due to the slope \( \alpha \) since the slopes are going to be relatively small on a highway, and to have \( F_r \) constant would allow us to precompute it.

The gravitational force is dependent on the slope as:

\[
F_g(s) = m g_r \sin(\alpha(s)),
\]

where \( m \) is the vehicle mass, \( g_r \) is the gravitational acceleration, \( \alpha(s) \) is the road slope in degrees, depending on the position \( s \).

In reality, many factors, including wind, road conditions and temperature are going to affect the drag forces acting on the HDV. Therefore a correction force, \( F_{err} \), is added to make sure these factors are taken into consideration.

This correction force is already being calculated by other cruise controllers in Scania trucks, so the model predictive controller is going to use it as an input. All together these forces form the total drag \( F_d \):

\[
F_d(v, s) = F_a(v) + F_r + F_g(s) + F_{err},
\]

(2.3)
2.3 Dynamics of the truck

Using Newton’s second law of motion and Equations (2.1), (2.2) and (2.3), we can write down the dynamics of the truck:

\[
\frac{d}{dt} \begin{bmatrix} s \\ v \end{bmatrix} = \begin{bmatrix} v \\ (F_e(T_e, T_b, v, g) - F_d(v, s))/m \end{bmatrix}
\] (2.4)
Chapter 3

Optimization problem statement

The optimization problem for fuel optimal truck driving can be formulated as a minimization problem where the fuel consumption \( M_f \) is to be minimized given a maximum trip time \( T_0 \).

\[
\begin{align*}
\text{minimize} & \quad M_f \\
\text{subject to} & \quad T \leq T_0, \\
& \quad \text{vehicle dynamics satisfied.}
\end{align*}
\]

It is possible to control engine torque, brake torque, and gear shift. Constraints on velocity and control signals may also be included in the problem statement. By relaxing the constraint on time, the problem can be reformulated as:

\[
\begin{align*}
\text{minimize} & \quad M_f + \beta T \\
\text{subject to} & \quad \text{vehicle dynamics satisfied.}
\end{align*}
\]

Here \( \beta \) represents a trade-off between fuel consumption and trip time. With this formulation (the same as [Monastyrsky and Golownykh (1993)]), it is not necessary to introduce time as a variable, but there is a need to tune \( \beta \) so that the desired trip time is achieved.

This chapter will continue with the description of two models that can be used to solve the optimization problem. The last one, “Fuel model” is the formulation that will be employed in the cruise controller.

3.1 Propulsive work model

In the beginning phase of the project, fuel consumption was considered to be linearly dependent on the propulsive work necessary to move the truck forward, as was assumed in paper [Hellström et al. (2010)]. Newton’s second law of motion states:

\[
m \frac{dv}{dt} = F_e(T_e, T_b, v, g) - F_d(v, s)
\]

This can be expressed in spatial coordinates as:

\[
m u \frac{dv}{ds} = F_e(T_e, T_b, v, g) - F_d(v, s).
\]
The propulsive work equals:

\[ W = \int_0^S F_e \, ds = \int_0^S \left( m \frac{dv}{ds} + F_d(s,v) \right) \, ds \]

\[ = \frac{m}{2} (v(S)^2 - v(0)^2) + \int_0^S F_d(s,v) \, ds. \]

Defining the kinetic energy at position \( s \) as \( e(s) = \frac{1}{2}mv(s)^2 \), the propulsive work can be written as:

\[ W = e(S) - e(0) + \int_0^S F_d \left( s, \sqrt{2e/m} \right) \, ds. \]

The time cost can be expressed as:

\[ T = \int_0^S \frac{ds}{v(s)} = \int_0^S \frac{ds}{\sqrt{2e/m}}. \quad (3.1) \]

The total cost will then be:

\[ J_{\text{total}} = e(S) - e(0) + \int_0^S \left( F_d \left( s, \sqrt{2e/m} \right) + \frac{\beta}{\sqrt{2e/m}} \right) \, ds. \quad (3.2) \]

The system dynamics can also be expressed in spatial coordinates, making velocity and current gear the only states. Velocity can be expressed in terms of kinetic energy, creating a model that uses energy and current gear as states:

\[ mvdv \, ds = \frac{m}{2} dsv^2 = \frac{de}{ds} = F_e \left( T_e, T_b, \sqrt{2e/m}, g \right) - F_d \left( \sqrt{2e/m}, s \right) = f(T_e, T_b, \delta g, e, s, g). \quad (3.3) \]

Gear-shifting can happen at any point in space and is represented by the control variable \( \delta g \). The exact modeling of a gearshift is described in Chapter 4.3.2, "gear-shift modeling".

The optimization problem using this approach can be written as:

\[
\begin{align*}
\text{minimize} & \quad W + \beta T \\
\text{subject to} & \quad \frac{de}{ds} = f(T_e, T_b, \delta g, e, s, g), \\
& \quad e_{\text{min}} \leq e \leq e_{\text{max}}, \\
& \quad T_{e,\text{min}}(e) \leq T_e \leq T_{e,\text{max}}(e), \\
& \quad 0 \leq T_b \leq T_{b,\text{max}}, \\
& \quad g \in \mathcal{G},
\end{align*}
\]

where \( \mathcal{G} \) is the set of allowed gears and \( \delta g \) is the control on the gearshift.

### 3.1.1 Calculation of the penalty parameter \( \beta \)

The choice of \( \beta \) is going to be crucial for the mean speed the vehicle is going to keep. A high value of \( \beta \) would give a high mean speed, and a low value of
\( \beta \) will give a low mean speed: to select a value of \( \beta \) so that the vehicle will keep the right speed can be a difficult task. Depending on the topology of the road ahead, the required fuel consumption to keep a certain mean speed will be different, and a single beta that ensures the same mean speed in the optimal solution for all roads is theoretically impossible.

The method chosen to work around this problem is to study a stationary solution to the optimization Problem (3.4). Assume that a gear is engaged, the slope is constant, and there exists one applicable control \( \hat{T} = T_e i(y) - T_b \) such that the vehicle keeps a constant velocity \( \hat{v} \), i.e. \( dv/dt = 0 \), and consequently \( de/ds = 0 \).

By Equations (2.3) and (2.4):

\[
\hat{T} = F_d(\hat{v},s) r_w. 
\]

Equation (3.2), describing the total cost, then becomes:

\[
\hat{J}_{\text{total}}(\hat{v}) = \int_0^S \left( \frac{F_d(\hat{v},s)}{c} + \frac{\beta}{\hat{v}} \right) ds. 
\]

The minimum can be found at the stationary point:

\[
\frac{\hat{J}_{\text{total}}(\hat{v})}{d\hat{v}} = \int_0^S \frac{d}{d\hat{v}} \left( \frac{F_d(\hat{v},s)}{c} + \frac{\beta}{\hat{v}} \right) ds 
= \int_0^S \left( \frac{d}{d\hat{v}} \left( \frac{F_d(\hat{v})}{c} \right) - \frac{\beta \hat{v}^2}{c} \right) ds 
= 0. 
\]

Solving for beta gives:

\[
\beta = \hat{v}^2 \frac{d}{d\hat{v}} \left( \frac{F_d(\hat{v})}{c} \right). 
\] (3.5)

This value of \( \beta \) can be interpreted as the value that gives mean velocity \( \hat{v} \) if the slope is constant, no gear-change is allowed, and there exists a control that satisfies the constraints and gives constant velocity. They are all restrictive assumptions, but it turns out in tests that the value works well for different conditions, even when Eco-roll is considered.

### 3.2 Fuel model

It turns out that considering only the propulsion work was not enough to obtain optimal solutions that were better than the solutions provided by a baseline modern look-ahead speed controller. The reason was that although this model would minimize the total work, the fuel consumption is not exactly linearly dependent on the produced work because the combustion efficiency is different at different operating points in the fuel map. Therefore a model was created using a look-up table for calculating the fuel consumption, \( f(T_e, v) \), as seen in Figure 2.2. Later the fuel consumption is transformed to energy by using an energy equivalent constant, \( \gamma \). This allows us to use a very similar model to (3.4), just with a more correct estimation of the energy/fuel consumption.

\[
W_{\text{fuel}} = \int_0^T f(T_e, v) \gamma dt = \int_0^S \frac{f(T_e, v) \gamma}{v} ds. 
\] (3.6)
As a result the final optimization problem will be:

\[
\minimize_{T_e, T_b, g} \ W_{\text{fuel}} + \beta T
\]

subject to

\[
\frac{de}{ds} = f(u, x),
\]

\[
0 \leq s \leq S,
\]

\[
e_{\text{min}} \leq e \leq e_{\text{max}},
\]

\[
g \in \{g_0, g_1, \ldots, g_D\},
\]

\[
T_{\text{emin}}(e) \leq T_e \leq T_{\text{emax}}(e),
\]

\[
0 \leq T_b \leq T_{\text{bmax}},
\]

(3.7)

where

\[
\begin{bmatrix}
    s \\
    e \\
    g
\end{bmatrix} = \begin{bmatrix}
    x
\end{bmatrix},
\]

\[
\begin{bmatrix}
    T_e \\
    T_b \\
    \delta g
\end{bmatrix} = \begin{bmatrix}
    u
\end{bmatrix}.
\]

The states, grouped in vector \( x \), are the position \( s \), the kinetic energy \( e \), and current gear \( g \).

The number of allowed gears will be \( (D + 1) \) as can be seen in the definition.

It is worth noting that \( v \) is a function of the state variable \( e \), and hence is not introduced as a separate state.

The control inputs, grouped in vector \( u \), are the flywheel torque \( T_e \), the brake-torque \( T_b \) and the gearshift \( \delta g \).

The estimation of beta treated in the previous chapter could still be used in this model, since the values for \( W \) and \( W_{\text{fuel}} \) are close to each other at many points, with just small variations due to the efficiency mapping of the engine. In practice the beta calculated in Equation (3.5) will be used for warm starting the problem, and the value will be updated every time the program runs again to make the next results better. The procedure is described in detail in Section 4.3.3 “Calculation of the trajectory”.

14
Chapter 4

Dynamic programming solution

Since the conditions may change during the drive mission due to disturbances, like delays due to traffic, it is important to compute new optimal solutions to Problem (3.7) during the drive mission. An efficient approach is to consider only a truncated horizon in each optimization: this will give an approximate solution to the problem with an accuracy that depends on the length of the horizon. Discretizing position, energy and torques allow us to use dynamic programming to solve the truncated problem.

4.1 Discretization

The optimization Problem (3.7) is discretized, both with respect to the control and the states. The entire drive mission will be divided into \( K \) evenly spaced steps, which are called the stages of the problem.

The energy \( e \) will be divided into \( Z \) steps; the control \( T_e \) will be divided into \( Q \) evenly spaced steps. \( g \) is integer valued and will be kept as it is. Braking is never going to be optimal, so it is not discretized.

After this discretization, the Problem (3.7) can be rewritten as:

\[
\begin{align*}
\text{minimize} & \quad J_1^*(x_1) = \xi_K(x_K) + \sum_{k=1}^{K-1} \xi_k(x_k, u_k) \\
\text{subject to} & \quad e \in \{e_1, e_2, \ldots, e_Z\}, \\
& \quad g \in \{g_0, g_1, \ldots, g_D\}, \\
& \quad T_e \in \{T_{e,1}, T_{e,2}, \ldots, T_{e,Q}\}, \\
& \quad 0 \leq T_b \leq T_{b,\text{max}}, \\
\end{align*}
\]

(4.1)

where

\[
\begin{align*}
x_k &= \begin{bmatrix} s_k \\ e_k \\ g_k \end{bmatrix}, \\
u_k &= \begin{bmatrix} T_{e,k} \\ T_{b,k} \\ \delta g_k \end{bmatrix}.
\end{align*}
\]
Here $\xi_k$ and $\xi_M$ defines the step cost and the terminal cost, respectively. They will be further described in the following chapters.

Every stage has $(D + 1)Z$ states, and we would have $(D + 1)ZK$ states in total if the whole mission was discretized. A visualization of this discretization can be seen in Figure 4.1.

4.2 Receding horizon

Since $K$ could be huge, the approach taken here is to look at a smaller horizon $N \leq K$ for the online optimization. This is done by approximating the cost-to-go at stage $N$. Problem (4.1) can be rewritten as:

$$J^*_1(x_1) = \min_{u_1, \ldots, u_{K-1}} \left\{ \sum_{k=1}^{N-1} \xi_k(x_k, u_k) + \min_{u_N, \ldots, u_{K-1}} \left\{ \xi_K(x_K) + \sum_{k=N}^{K-1} \xi_k(x_k, u_k) \right\} \right\}.$$ 

Define the residual cost

$$J^*_N(x_N) = \xi_K(x_K) + \sum_{k=N}^{K-1} \xi_k(x_k, u_k)$$

as the cost-to-go function at stage $N$. This function will be replaced by an approximation $\tilde{J}_N(x_N)$. The objective function is now only defined in the look-ahead horizon as

$$\min_{u_1, \ldots, u_{N-1}} \left\{ \tilde{J}_N(x) + \sum_{k=1}^{N-1} \xi_k(x_k, u_k) \right\}.$$
4.3 Dynamic programming Algorithm

Denote by $U_k$ the set of allowed controls and by $X_k$ the set of allowed states at stage $k$. The DP solution to the look-ahead problem is described in Algorithm 1.

Algorithm 1 Dynamic programming algorithm for general look-ahead problem

```plaintext
for all $x \in X_N$ do
    $J_N(x) \leftarrow \tilde{J}_N(x)$
end for

$k \leftarrow N - 1$
while $k > 0$ do
    for all $x \in X_k$ do
        $J_k(x) \leftarrow \min_{u \in U_k} \{ \xi_k(x, u) + J_{k+1}(F_k(x, u)) \}$
    end for
    $k \leftarrow k - 1$
end while

Output: The policy with the optimal cost $J^*_1(x_1) = J_1(x_1)$.
```

The basic principle in Algorithm 1 is that if the costs-to-go $J_l(x)$ are known for every state at stages $l$, $l \geq n$, then the costs-to-go $J_{n-1}(x)$ can be computed as a function of $J_l(x)$, $l \geq n$.

This principle is applied to Problem 4.1. Assuming all the costs-to-go for states $\{(s_l, e_z, g_i) \mid l \in \{n, \ldots, N\}, z \in \{1, \ldots, Z\}, d \in \{0, \ldots, D\}\}$ are known the costs-to-go for every state $\{(s_{n-1}, e_z, g_d) \mid z \in \{1, \ldots, Z\}, d \in \{0, \ldots, D\}\}$ without gear shifting is calculated. The results are denoted by $J_{cs}(s_{n-1}, e_z, g_d)$.

After that, gear shifts are considered and the costs-to-go with gear shift, $J_{gs}(s_{n-1}, e_z, g_d)$ are calculated. Finally, the cost-to-go at each state is given by

$$J(s_{n-1}, e_z, g_d) = \min\{J_{cs}(s_{n-1}, e_z, g_d), J_{gs}(s_{n-1}, e_z, g_d)\}.$$ 

The expressions for the cost-to-go in each respective case are derived in the following chapters.

When the cost-to-go has been calculated for every state $\{(s_n, e_z, g_i) \mid n \in \{1, \ldots, N\}, z \in \{1, \ldots, Z\}, d \in \{0, \ldots, D\}\}$, it is possible to start the algorithm from the current velocity and position of the truck and calculate an optimal trajectory by selecting at every stage the option with lower cost-to-go.

4.3.1 Constant gear modeling

Consider a vehicle at state $(s_n, e_{prev}, g)$. To calculate the cost-to-go, we will need to calculate the step-cost for the step from $s_n$ to $s_{n+1}$ and add to it the cost-to-go at position $s_{n+1}$. It is necessary to consider all possible applicable torques ($T_e \in \{T_{e,1}, T_{e,2}, \ldots, T_{e,Q}\}$) and add to them the cost-to-go at point $(s_{n+1}, e_{next}, g)$ where $e_{next}$ is the kinetic energy at position $s_{n+1}$, if a certain specific torque is applied starting at $(s_n, e_{prev}, g)$. The energy at next stage is calculated using Equation (3.3) solved using Euler forward, for a distance of $h = s_{n+1} - s_n$. To get a precise estimation of the cost function, it is necessary
to know the velocity at next stage. Velocity is a function of kinetic energy, but to obtain the velocity from the kinetic energy there is a need to use the square root operator. This operation is costly on a computational level, and since the objective is to have a fast program that has to run on an embedded controller it was chosen to avoid the use of square root altogether. Therefore, instead of using Euler forward on the energy and then using the square root to find the velocity it was chosen to apply Euler forward to both energy and velocity. This would give a small difference in the resulting velocity compared to taking the square root of the energy, but that difference is marginal. The algorithm, based on Equation (3.3) is:

\[
e_{\text{next}} = e_{\text{prev}} + hf(u, e_{\text{prev}}, s, g) = e_{\text{prev}} + \delta e,
\]

\[
v_{\text{next}} = v_{\text{prev}} + \frac{h}{v_{\text{prev}}m} f(u, e_{\text{prev}}, s, g) = v_{\text{next}} + \frac{\delta e}{v_{\text{prev}}m}.
\]

To calculate the step-cost, the objective function has to be integrated in the interval \([s_n, s_{n+1}]\). The velocity in the interval is approximated to be the mean between the velocity at \(s_n\) and the velocity at \(s_{n+1}\):

\[
v_{\text{mean}} = \frac{v_{\text{prev}} + v_{\text{next}}}{2}.
\]

The energy used during this step is calculated based on Equation (3.6):

\[
\Delta W(e, g, u) = \frac{f(T_e, v_{\text{mean}}) \gamma h}{v_{\text{mean}}}.
\]

The time taken is calculated based on Equation (3.1)

\[
\Delta T(e, u) = \frac{h}{v_{\text{mean}}}.
\]

The step-cost is then:

\[
\xi(e, g, u) = \Delta W(e, g, u) + \Delta T(e, u) \beta.
\]
The cost-to-go at position \( s_{n+1} \) and energy \( e_{\text{next}} \) is given by linear interpolation of \( J(s_{n+1}, e_{k-1}, g) \) and \( J(s_{n+1}, e_k, g) \) where \( e_{k-1} \leq e_{\text{next}} \leq e_k \). (see Figure 4.2). The interpolated value is denoted by \( \tilde{J}(u_k) \). The cost-to-go without gear shift at position \( s_n \) and energy \( e_{\text{prev}} \) is obtained by finding the control signal \( u_k \) that minimizes the sum of the cost-to-go at position \( s_n \) and the step cost:

\[
J_{cg} = \min_{u_k} \{ \tilde{J}(u_k) + \xi(e, g, u) \}.
\]

Braking is never optimal and will be considered only if \( e_{\text{next}} \) exceeds the maximum kinetic energy limit. In this case, the only choice will be to brake with a brake torque \( T_b \) so that \( e_{\text{next}} = e_{\max} \) with \( e_{\max} \) being the maximum allowed kinetic energy.

### 4.3.2 Gear shift modeling

Gear shifting requires slightly more advanced modeling. It is supposed that the change from \( g_{\text{prev}} \) to \( g_{\text{next}} \) takes a constant time \( t_{\text{shift}} \), in which the vehicle is going to roll freely. It is important to estimate how long the vehicle is going to move during this time, and therefore we use the dynamics of Equation (2.4). The driving force during the shift is zero, and the time taken is \( t_{\text{shift}} \), using Euler forward:

\[
v_{\text{next}} = v_{\text{prev}} + t_{\text{shift}} - \frac{F_d(v, s)}{m}.
\]

\( v_{\text{next}} \) can be transformed into \( e_{\text{next}} \). The driven stretch of road becomes then:

\[
\Delta S = \int_{v_{\text{prev}}}^{v_{\text{next}}} v \delta t \approx t_{\text{shift}} v_{\text{prev}} + \frac{v_{\text{prev}} + v_{\text{next}}}{2}.
\]

Now the energy consumption for the gear-change can be calculated using the same fuel factor table as for the constant gear. The torque is calculated as the torque necessary to change the rotational speed of the crankshaft from the start of the gear-change to the end of it. These rotational speeds can be calculated from the start and end velocity using Equation (2.2). Having rotational speed and torque will allow us to calculate the amount of fuel that should be consumed in the gear-change and from that the energy loss can be calculated with \( \gamma \).

The work done by the engine to change speed will then be:

\[
T_{\text{change}} = \frac{(\omega_{\text{next}} - \omega_{\text{prev}})J_e}{t_{\text{shift}}},
\]

\[
\Delta W = f(\frac{T_{\text{change}}, \omega_{\text{next}} + \omega_{\text{prev}}}{2}) \gamma t_{\text{shift}}.
\]

The step cost is then:

\[
\xi(e, g, u) = \Delta W(e, g, u) + t_{\text{shift}} \beta.
\]

The cost-to-go at position \( s_n + \Delta S, e_{\text{next}} \) is given by bilinear interpolation between the 4 costs around this position: \( J(s_{p+1}, e_{k+1}, g_{\text{next}}), J(s_{p+1}, e_k, g_{\text{next}}), J(s_p, e_{k+1}, g_{\text{next}}), J(s_p, e_k, g_{\text{next}}) \), see Figure 4.3. Here \( s_p, s_{p+1}, e_k, e_{k+1} \) are discretization points so that \( s_p \leq s_n + \Delta S \leq s_{p+1} \) and \( e_k \leq e_{\text{next}} \leq e_{k+1} \). The cost-to-go with gearshift at position \( s_n \) and energy \( e_n \) is obtained by finding the control signal \( \delta g \) that minimizes the sum of the cost-to-go at position \( s_n \) and the step cost.
4.3.3 Calculation of the trajectory

As introduced at the beginning of this section, after the costs-to-go are calculated for the whole discretized horizon it is possible to run the algorithm in a forward manner, starting from the current velocity and position and calculating the option with the lowest cost-to-go from there to the next stage. Choosing this option will give you a velocity in the next stage, and, by doing the same thing at this new stage, a complete trajectory can be drawn inside the matrix of discretized states, having a velocity at each stage and a strategy between the different stages. The first sample is then going to get applied as a control, and the rest of the trajectory is going to be used to check that the value of $\beta$ we are using is correct.

As introduced in the previous chapters, $\beta$ can be evaluated for constant slopes when the fuel consumption is assumed to be linearly dependent on propulsive work. In general, this estimation gives a good value for $\beta$, but the actual value of $\beta$ would be unique for every road profile, and the nonlinearities added by Eco-roll and the fuel map will demand a different value than the one calculated to keep the desired mean speed consistently.

A way to workaround this problem is to use a controller on $\beta$, and base the control signal on the estimated speed in the road ahead. That would allow us to continuously update the value of $\beta$ to face new stretches of road before the truck has even driven through them, giving a very fast control. If the mean velocity in the horizon is different from the wanted mean velocity a quadratic term is going to be added to this value so that hopefully next time the results will be better. It was chosen to use quadratic control since it is very effective at correcting large deviations from the desired value, and this is exactly what

![Figure 4.3: Cost-to-go after a gearshift.](image)
is desired, small deviations are always going to exist and are not as important. The formula for it will be:

\[
\beta_{\text{new}} = \beta_{\text{old}} + (v_{\text{wanted}} - v_{\text{mean}})^2 \cdot \eta \cdot \text{sign}(v_{\text{wanted}} - v_{\text{mean}})
\]

\( \eta \) is a scaling factor, and the sign is added so that this change can be both positive or negative. This dynamic adjustment of beta is going to help keep the right mean velocity at all times, and no test have given a mean speed which is more than 1 km/h different from the wanted mean speed.

### 4.3.4 Handling differences in torque

One of the problems of this formulation is that every step does solve a new optimization problem, without regards to the solution that was used in the step before. This could create some problems since there are parts of the fuel consumption table that are very close in efficiency but quite far when it comes to supplied torque. So if the optimization is let free, it could “jump” between optimal torques that are situated far from each other, as shown in Figure 4.4.

This is not good for two reasons:

- It might not feel good while driving since you would feel a different acceleration every second.
- The actual supplied torque will be different from the calculated torque in the models, since when a new torque gets requested the engine will need some time to change from the torque it has now to the requested one. In the model, on the other end, these torque changes are assumed to be instantaneous.

To solve this problem, a cost for changing the torque is introduced in the last step of the optimization, where the trajectory is calculated. It is going to be added here since optimizing for it when the costs-to-go are calculated would require an additional state, the current applied torque, and therefore slow down computations significantly. The cost added will be quadratic and aspires at minimizing the difference from the previous supplied torque. This choice is dictated by the fact that it is large differences in supplied torque that want to be avoided, not small ones. The formula for it will be:

\[
C_{\text{torque}} = (T_{\text{prev}} - T_e)^2 \epsilon_t \text{time}, \quad (4.2)
\]

\( T_{\text{prev}} \) is the torque used in the step before, \( T_e \) is the torque to be applied now and \( \epsilon_t \) is a factor, typically small since the objective is to not change the result too much. The cost function used when calculating the optimal trajectory will be:

\[
C_{\text{total}} = W_{\text{fuel}} + \beta T + C_{\text{torque}}.
\]

This implies that the cost function used when calculating the trajectory is not the same as the cost function defined in (3.7), and used to calculate the cost-to-go at every stage.

If the vehicle is shifting from neutral to another gear, \( T_{\text{prev}} \) would be zero, but it is desirable to apply a higher torque quickly. Therefore it is not suggested to use Equation (4.2) as it is since it will force the algorithm to rise slowly in
torque demand. Initially, it was decided not to apply any extra cost when shifting from Eco-roll, but the results from the first road test showed that an essential element had been forgotten: the turbo boost. In Figure 4.5, the actual gear change process from road tests can be analyzed. The red curve represents the actual torque of the vehicle during the gearshift. By looking at this curve we can see that when the shift begins, around second 1.7, a torque gets applied to speed up the engine to the cruising speed. Afterward the torque comes down to zero again and later speeds up very fast up to around 1000 Nm. The whole process takes about one second (which confirms that the model for gear-change presented earlier was quite realistic) but the interesting part is after that: even though the requested torque is high, the actual torque rises very slowly. This happens due to the turbocharger lag: when the vehicle was in neutral the engine was working under the boost threshold of the turbocharger, and now the engine response is slow. The model does not take the turbo into consideration, but it was chosen to add a cost for requesting a torque higher than 1100 directly after a gear-change, which makes the requested torque slightly more realistic, as seen in the figure for the second road test in Figure 4.5. This cost is also added only in the trajectory, as was the cost for changing torques described in Equation (4.2).
Figure 4.5: Figure depicting the gear changing process when requiring a high torque. The blue curve represents the requested torque, and the red curve represents the actual engine torque. The green curve represents the speed of the engine, going from 500 (neutral) to rolling speed. In the second test, a price for putting torques over 1100 directly after a gear-change from neutral was added.

4.3.5 Handling interpolation at boundaries

Keeping the speed of the vehicle within given limits is important. This is achieved by running the dynamic programming algorithm only between specific allowed speeds but results in the problem of how to handle the cost at boundaries.

Starting from a certain state (speed and gear), if it is impossible to find a control that reaches an allowed state in the next stage, the cost-to-go at the starting state has to be set to infinity, making this point infeasible and preventing the vehicle from reaching that point. It could though happen that a certain control brings the vehicle to a kinetic energy which lays between a feasible and infeasible state. In Figure 4.6 this situation is exposed: starting from state \((e_{\text{prev}}, g, s_n)\) a specific control strategy would bring the vehicle to the point \((e_{\text{next}}, g, s_{n+1})\).

To get the cost-to-go at this point, it would be necessary to interpolate between \(j(s_{n+1}, e_k, g)\) and \(j(s_{n+1}, e_{k+1}, g)\), but \(j(s_{n+1}, e_{k+1}, g) = \infty\)!

A natural solution is to give infinity as a cost-to-go in this point, but this would make Point \((e_{\text{prev}}, g, s_n)\) infeasible, which is not always true since \(e_{\text{next}}\) is an allowed kinetic energy. This strategy could potentially make the whole solution infeasible when in practice it is feasible.

Another solution could be to extrapolate between the cost-to-go of the two nearest feasible points to the arrival point. This solution could also be dangerous since it could give a low cost-to-go to a state for which the only possible next step will be infeasible, and lead the program to pursue an infeasible solution when there are good available feasible ones.

The solution adopted here is to add a “penalty” cost to the interpolation when one of the points interpolated between is infeasible. This penalty, \(\Omega\) in the figure, should be big enough to avoid that the program chooses this solution unless it’s the only option available. In this case, the program can still lead to an infeasible solution, but it is guaranteed that it will not do so if there is a
feasible one available. A typical example when it is useful is if there is a short steep stretch of road. If the truck arrives with high speed it is possible for it to manage the whole uphill without going off the boundaries, but if the truck arrives with a low speed it will not manage to keep the speed up. In the unfortunate case that the program should converge but lead to an infeasible solution this would be shown in the trajectory and preventive measures can be taken.

4.4 Memory requirements

The algorithm was developed in C following Scania coding guidelines. Most of the numerical values were saved as floating point values, and some of them were saved as 8-bit or 16-bit unsigned integers.

The heart of this algorithm is the array in which all the costs are stored. This array’s size will depend on the number of gears analyzed (\(D + 1\), \(D\) being the number of gears without Eco-roll), the number of discretization steps in the kinetic energy (\(Z\)) and the number of steps forward taken (\(N\)). The size of this array will be: \((D + 1) \cdot Z \cdot N\).

Also, it is necessary to save the array keeping track of the fuel consumption over the whole working range of the engine (Figure 2.2). The engine speed span between 500 and 2400 RPM, discretizing it at one point every 10 RPM gives us 191 points. Since the program is only going to use \(Q\) discretized torques, we only need to get the fuel consumption at these levels. Therefore the array
that will be saved will have size: \(191 \cdot Q\). The fuel consumption values range between 0 and around \(20\ [\text{g/s}]\), so it works well to multiply all the values by 1000 and save this values at 16-bit unsigned integers: the range will then be between 0 and \(20000\ [\text{mg/s}]\). By using this trick, the array will take up half the space compared to saving all the values as floating points, with minimal loss of precision.

As the program is coded right now, there is a function calculating the index of maximum and minimum allowed torque at every speed. This function uses three floating points vectors as large as the engine speed discretization, 191 elements each. There are also several other smaller vectors and matrices; the largest ones will be shown here:

- 3 vectors of size \(N\) and 3 vectors of size \(N - 1\) (keeping track of the trajectory: slope, speed, gear, flywheel torque, brake-torque, drag vector from gravity)
- 4 vectors of size \(Z\), (discretized speeds in \(\text{km/h}\), discretized speeds in \(\text{m/s}\), discretized kinetic energies, vector containing the drag at different speed)
- 1 vector of size \(Q\) containing the torques
- 2 matrix of size \(Z \times D\) containing the max and min index of the torques utilizable at different speeds (saved as 8-bit unsigned integers).
- 4 vectors of size \(D + 1\) keeping gear related values
- less than 100 other floating point and unsigned integers variables.

The total size of the program in bytes (\(\Theta\)) will then be:

\[
\Theta < ((D+1) \cdot Z \cdot N + 191 \cdot 3 + 6N + 4Z + Q + 4 \cdot (D+1) + 100) \cdot 4 + Z \cdot D + 2 \cdot 191 \cdot Q
\]

The version of the program that was tested on a HDV had \(Q = 20\), \(Z = 22\), \(D = 1\), \(N = 60\) giving a size less than 23 kB.
Chapter 5
Simulation Results

5.1 Simulink® simulation environment

Scania utilizes a simulation environment coded in Simulink to test the performance of their cruise controllers. Many simulations were performed in this environment to test the behavior of the algorithm on a realistic road profile. The reported tests were performed on a simulation of the stretch of road between Södertälje and Norrköping, and specifically the first 100km starting from Södertälje. The stretch of road is appropriate for this type of controller since it has a lot of small variations in height, but no slopes that are big enough to force the trucks to slow down to velocities under 70.4 km/h. It was chosen to simulate a truck weighing 29 ton with overdrive automatic gearbox and 3.07 rear gear ratio, as the truck used in the trial run (Table 6.1). It was decided to keep 80 km/h as a reference speed with the possibility to accelerate up to 85 km/h and decelerate down to 70.4 km/h (12 % less). Also, it is possible to accelerate up to 87 km/h for very short periods of time.

Figure 5.1: The stretch of road between Södertälje (A) and Norrköping (B). Map from Google®

5.2 Simulations

Three simulation results are going to be reported: “Test A” involving one gear and Eco-roll, “Test B” involving three gears but with fewer discretization points
for the kinetic energy, and “Test C” with three gears and the same amount of discretization points for kinetic energy as “Test A”. The parameters can be seen in Table 5.1. Results can be seen in Table 5.2. The performance of the tests is compared with a with a baseline look-ahead speed controller which is rule based (CCRB).

### Table 5.1: The parameters for the reported tests.

<table>
<thead>
<tr>
<th>Test</th>
<th>N. of kinetic energy disc. points</th>
<th>N. of gears (including neutral)</th>
<th>N. of torque discretization points</th>
<th>N. of steps</th>
<th>Step length</th>
<th>Horizon</th>
<th>Max speed</th>
<th>Min speed</th>
<th>Desired mean speed</th>
<th>Memory requirements</th>
<th>Time to run on PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>22</td>
<td>2</td>
<td>20</td>
<td>60</td>
<td>40m</td>
<td>2400m</td>
<td>85 km/h</td>
<td>70.4 km/h</td>
<td>80 km/h</td>
<td>22,8 kB</td>
<td>0,004 s</td>
</tr>
<tr>
<td>B</td>
<td>11</td>
<td>4</td>
<td>20</td>
<td>60</td>
<td>40m</td>
<td>2400m</td>
<td>85 km/h</td>
<td>70.4 km/h</td>
<td>80 km/h</td>
<td>22.7 kB</td>
<td>0,0067 s</td>
</tr>
<tr>
<td>C</td>
<td>22</td>
<td>4</td>
<td>20</td>
<td>60</td>
<td>40m</td>
<td>2400m</td>
<td>85 km/h</td>
<td>70.4 km/h</td>
<td>80 km/h</td>
<td>33.5 kB</td>
<td>0,0113 s</td>
</tr>
</tbody>
</table>

### Table 5.2: The results from three tests on the Södertälje-Norrköping road.

<table>
<thead>
<tr>
<th>Test</th>
<th>Mean velocity</th>
<th>Fuel consumption</th>
<th>Time in Eco-roll</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCRB</td>
<td>79.86 km/h</td>
<td>0,2410 l/km</td>
<td>1170 s</td>
</tr>
<tr>
<td>CCMP - A</td>
<td>79.78 km/h (-0.10%)</td>
<td>0,2392 l/km (-0.74 %)</td>
<td>1684 s</td>
</tr>
<tr>
<td>CCMP - B</td>
<td>79.78 km/h (- 0.10%)</td>
<td>0,2384 l/km (-1.07 %)</td>
<td>1705 s</td>
</tr>
<tr>
<td>CCMP - C</td>
<td>79.81 km/h (- 0.07%)</td>
<td>0,2381 l/km (-1.19 %)</td>
<td>1705 s</td>
</tr>
</tbody>
</table>

5.2.1 Analysis of the driving strategy

To understand how the vehicle drives in the different simulations it is very helpful to look at Figures 5.2, 5.3, and Figure 5.4.

In Figure 5.2 a short stretch of the simulation when driving with a single gear (Test A) is presented. The figure shows velocity, flywheel torque and gear choice for the CCRB controller and the CCMP controller with a single gear choice. Also, the elevation of the simulated road is presented.

It can be seen that when facing a long steep downhill section (between km 8 and 10) the behavior of the two controllers is very similar: lower the speed as much as possible before the downhill and then roll up to the maximum allowed speed. This is one of the primal function of both controllers: avoid unnecessary use of brakes. By lowering the speed before a long downhill stretch, the amount of energy that gets lost in braking diminishes and consequently less fuel is wasted. The behavior is different when the slopes are more moderate. The CCRB seems to keep the desired speed, 80, most of the time whereas the CCMP, in general, tends to take advantage of every small downhill section and run short accelerations periods followed by small periods in neutral gear, Eco-roll. Therefore the speed is rarely exactly 80 km/h but shifts between 76 and 83. This behavior allows the controller to use points in the fuel map that are more efficient for the
acceleration stretches (since it does not need to keep a certain velocity), and to Eco-roll for longer time compared to the CCRB. The clearest examples of this strategy happen around km 1.5 and 5 in Figure 5.2. Here the CCMP avoids using the low torques in the downhill section since the engine is not efficient at these torques. Instead, it chooses to Eco-roll: it is going to pick up speed later applying more even torque.

In Figure 5.3 the same stretch of road is simulated using the parameters of Test C. Now the CCMP is allowed to shift gear and uses this possibility to drive even longer sections in Eco-roll, as can be seen between km 1-2 and 4-6. The peak at km 2.5 is entirely avoided by using a lower gear. Also, this Simulation drives a little bit faster at the end of the segment, shifting gear in a similar way to the CCRB. It looks like this simulation focuses even more on specific flywheel torques, and this is very clear when looking at Figure 5.4.

In Figure 5.4 the torque outputs are plotted inside the fuel map, giving a clear idea of where in the engine fuel map each simulation drives most time. In the CCRB simulation, two parallel lines of blue dots can be seen. Each line represents points with the same engine speed, and all the dots represent different torques. Clearly, this solution is keeping a constant speed and using two gears (direct gear and overdrive). The direct gear is the gear with a transmission ratio of one between engine speed and driveshaft speed: it is the second highest available gear and it is the gear with the lowest transmission losses. The overdrive gear is the gear with a transmission ratio less than one between engine speed and driveshaft speed: it is the highest available gear and it assures the lowest possible engine speed given a specific vehicle speed, minimizing losses due to motor friction. Clearly, the left line corresponds to the overdrive gear and right one corresponds to the direct gear. The big blue circle at the origin represents the time the vehicle Eco-rolls, and the small circles with negative torque represent moments when the vehicle is rolling with the gear connected.

By looking at the results of CCMP Test A two interconnected behaviors can be observed: that the vehicle is driving at a broader spectrum of speeds compared to the CCRB, and that the torques are concentrated in specific points (where the engine is most efficient). Also, this simulation drives for a longer time in Eco-roll.

In Test B and C the range of values the flywheel torque assumes is even smaller, concentrated near a specific point for the direct gear and another point for the overdrive gear. The controller avoids using both very high torques and very low ones. Also, Eco-roll is used slightly more often in these tests.
Figure 5.2: 14 km of road simulation driving with the controller in Test A compared to the CCRB.
Figure 5.3: 14 km of road simulation driving with the controller in Test C compared to the CCRB.
Figure 5.4: The torque usage of the CCRB compared to the model predictive cruise controllers used in the tests. The blue circles represent the number of times a certain combination of flywheel torque and engine speed have been used by each simulation. The red curves represent maximum and minimum allowed flywheel torque.
5.2.2 Energy analysis

A graph depicting the energy losses of the three tests compared to the base case (CCRB) can be seen in Figure 5.5. By analyzing this figure, it is possible to understand how the three tests save fuel compared to the CCRB. It can be seen that the energy losses due to rolling and air resistance are similar for all the simulations and the CCRB. This fact is not surprising since every scenario had to keep the same mean speed. The transmission losses show that Test B and C saves a lot of energy compared to the CCRB and Test A loses energy. Tests B and C save energy because they use the direct gear far more than the CCRB, giving fewer transmission losses. The direct gear is not available for the single gear Test, and therefore it loses some energy there. The friction losses are also interesting: we can see that Test A saves a lot of energy but Test B and C do not. This happens because Test A Eco-rolls more than CCRB and does not use direct gear; therefore it consumes less to over-win the friction in the engine. Test B and C also use Eco-roll a lot, but due to the frequent use of the direct gear they also lose much of the saved energy to over-win engine friction when using this gear. It is also possible to see that in all the three tests the vehicle brakes a little bit more than when using the CCRB, probably due to a less precise breaking strategy. The potential energy is clearly the same, and some small difference can be seen in the kinetic energy at the end of the Test. It is now interesting to check the last two columns: the sum and the fuel consumption. It can be seen when looking at Test A that the total sum of the energy losses is smaller than for Test B and C, but when looking at the fuel consumption Test B and C did better, why? This happens because the points in which Test B and C apply their respective torques are more energy efficient when it comes to fuel consumption, and allows these tests to perform better even if the total energy losses, especially due to engine friction, are bigger. This result clearly shows why it is important to take information about the fuel map in account when optimizing for this problem.
5.2.3 Issues

Due to the discretization of the torque and the nonlinear nature of the fuel map, it is rare for the program to give out a perfectly even torque profile, there are always going to be small “jumps” as described in Section 4.3.4, and visualized in Figure 4.4.

This behavior is not so much of a problem when applying a style of driving characterized by accelerations followed by periods of Eco-roll, but a more traditional controller would give out a more even torque profile when keeping a constant speed on a flat road since the optimal torque for that could be between two discretization points.

The dynamics after a gear-change cannot be exactly modeled as shown in Figure 4.5. A possible solution could be to add torque as a state, but that would slow down computations significantly due to the “curse of dimensionality”.

Another behavior that is not necessarily optimal is what happens at km 8 in Figure 5.3. The stretch of road is shown in detail in Figure 5.6. I can be seen that the CCMP decides to first Eco-roll, bring the speed down to almost 70, then apply a small torque to keep the speed at 70.4 (the lower limit) until the downhill starts. Instead, the CCRB chooses to apply the drive torque for a bit longer, then begin slowing down using the engine fiction until the downhill part begins and the velocity starts to rise again.

Both solutions agree that it is optimal to have a velocity of 70.4 km/h at the beginning of the downhill and to use the friction from the engine to slow down the truck until maximal velocity is reached and the real brakes must be used. To reach this velocity only by rolling proves to be difficult for the CCMP, probably
due to the relatively large discretization used and because of the cost added for
interpolations near the boundaries (Section 4.3.5).

Eco-rolling, when the slope is positive, is always going to result in a velocity
that is lower compared to the starting velocity; therefore the cost of driving in
neutral gear exactly at the boundary will be infinite. This fact would produce
repercussions in the cost of driving in neutral gear near the boundary since there
is a penalty for interpolating when one cost is infinite as shown in Figure 4.6.
Specifically, the cost for Eco-rolling between the lowest discretization point and
the one over it will be very high. Therefore the choice will go towards shifting
to a gear when reaching this velocity, doing so it is possible to control the ve-
locity at the boundary and not get any points with infinite cost. Since the cost
for this little acceleration is relatively low, due to the lower air resistance at
70.4 km/h and the more favorable slope, this option seems to be cheaper than
the option used in the CCRB, at least when taking into consideration only the
discretized points (there is no extra cost in the CCMP for applying this same
strategy, but it could still be that having a more precise discretization would
show otherwise).

Probably small savings could be obtained by starting to use Eco-roll a little
bit later, and then switch to braking using engine friction without using low
torques, but to find this solution one would need a more precise discretization
or some methods not relying on discretization.
Figure 5.6: The behavior of the CCMP compared to CCRB when affronting longer slopes. The brake-torque is almost the same for both controllers, so only one line is plotted for clarity.
Chapter 6

Trial run

Two separate trials were performed on the E4 highway between Södertälje and Norrköping. The first trial was about one hour long and the second about half an hour. Both trials started and ended at Scania development center in Södertälje.

6.1 Test setup

<table>
<thead>
<tr>
<th>Bumblebee</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass</td>
<td>29 ton</td>
</tr>
<tr>
<td>Engine</td>
<td>DC13148/450hk</td>
</tr>
<tr>
<td>Gearbox</td>
<td>GRSO905R</td>
</tr>
<tr>
<td>Rear axle gear ratio</td>
<td>3.07</td>
</tr>
</tbody>
</table>

Table 6.1: Data of truck used in the trial run.

The data of the truck used for testing can be found in Table 6.1. This is one of Scania's test trucks, where it is possible to replace the gearbox controller software. The parameters for the algorithm are described in Table 6.2 and are the same as Test A, except for a shorter horizon. It was programmed to work in parallel with the code for the other software running on the ECU, including the current speed controller, a controller deciding the shifting strategy, and much more. The CCMP activated itself only between the allowed speeds (70.4 km/h to 85 km/h), having as inputs the current speed, the current gear, and the current slope of the road ahead. All this information can be obtained from variables and functions already coded in the gearbox ECU. The algorithm calculates an optimal driving strategy based on the current gear, having the possibility to shift only between this gear and Eco-roll. The output
<table>
<thead>
<tr>
<th>Test</th>
<th>Trial run</th>
</tr>
</thead>
<tbody>
<tr>
<td>N. of kinetic energy disc. points</td>
<td>22</td>
</tr>
<tr>
<td>N. of gears (including neutral)</td>
<td>2</td>
</tr>
<tr>
<td>N. of torque discretization points</td>
<td>20</td>
</tr>
<tr>
<td>N. of steps</td>
<td>60</td>
</tr>
<tr>
<td>Step length</td>
<td>25m</td>
</tr>
<tr>
<td>Horizon</td>
<td>1500</td>
</tr>
<tr>
<td>Max speed</td>
<td>85 km/h</td>
</tr>
<tr>
<td>Min speed</td>
<td>70.4 km/h</td>
</tr>
<tr>
<td>Desired mean speed</td>
<td>80 km/h</td>
</tr>
<tr>
<td>Memory requirements</td>
<td>22.8 kB</td>
</tr>
<tr>
<td>Time to run on PC</td>
<td>0.004 s</td>
</tr>
</tbody>
</table>

Table 6.2: The parameters for the trial run

of the algorithm will then be a suggested driving torque, and a Boolean variable deciding if the vehicle should Eco-roll or keep the current gear. Gear-shifting is left to the same software deciding gearshifts in commercially available Scania trucks, and when a gear shift occurs the model predictive controller will calculate a strategy for the new gear.

If the velocity is out of the boundaries, or the algorithm does not find a solution for other reasons, the cruise controller is going to fall back to the strategy suggested by previously installed software.

6.2 Results

The results showed a behavior similar to the simulations, with an alternation of Eco-roll and accelerations using an efficient torque.

During the first Test, the mean velocity when the CCMP was active was 79.4 km/h and during the second Test 79.1 km/h. These velocities are slightly lower than the wanted 80 km/h, but is not very surprising. During real driving there is going to be an unwanted use of brakes in certain situations, plus the torque supply is not always the same as the demanded torque (see Figure 4.5 for an example of the differences coming from the turbo lag). A possible solution would be to use a slightly higher reference velocity. In Figure 6.1 a part of the second Test drive is shown. The stretch of road is part of the road depicted in the simulation Figures 5.2-5.3.
Figure 6.1: The first 12.5 km of the E4 motorway from the Scania facility in Södertälje towards Norrköping. This is from the second Test, the mean velocity during this 12.5 km is almost exactly 79 km/h. This stretch of road is part of the road in Figure 5.2.
Chapter 7

Conclusions

In this thesis, a model predictive cruise controller based on Dynamic Programming was developed and tested. The whole controller has been developed following Scania coding guidelines for embedded software and was tested inside a production gearbox ECU on a Scania truck. Information about the performance of the controller was obtained by simulating its behavior in a Simulink environment. In this environment, it was shown that it is possible to achieve savings up to 1% compared to a rule-based look-ahead speed controller on the stretch of road between Södertälje and Norrköping. The savings were achieved mainly by using the neutral gear more often and alternate accelerating periods using an efficient torque to periods in neutral gear: driving in this way allows to use more efficient points in the fuel map for forward propulsion and lose less energy due to engine friction. This style of driving seems to be very advantageous on certain road profiles, but would be difficult to apply in heavy traffic and will pose some problems from a comfort point of view: more testing is suggested before adopting a controller of this type commercially. Nevertheless, it was shown that it is fully possible to solve relevant optimization problems on a production ECU of a commercial truck. Furthermore, the solution is stable and saves fuel on certain slopes.

7.0.1 Future work

The results from simulations using this controller could be used to evaluate the performance of commercial speed controllers. The fact that the code has been developed inside the same environment that is used for the development of Scania’s commercial speed controllers allows for easy comparison. Also the fact that the code can run on a gearbox ECU allows the developers to easily evaluate if the style of driving feels acceptable on an actual motorway. It is possible to continue the project in two directions:

- Towards an actual commercial application.
- As a benchmark for other controllers.

In case a version of the program is to be used commercially more work should be put into making the solution more reliable and capable of handling the most
different situations, maybe by communicating better with other controllers inside the ECU. Right now the controller is very limited by the speed constraints and the number of gears analyzed, so it needs more work to assure a pleasurable and efficient ride in all situations.

On the other line, if Scania decides to use the program as a benchmark, more functions could be added, maybe to take into account differences in torque, the turbocharger, temperature of exhaust gasses or other factors. Depending on how many functions are going to be added the possibility to test the program on an ECU could be lost, but new interesting driving strategies could be found.
Bibliography


