FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Motorcycle modeling for eCVT-in-the-loop real-time hybrid testing

Simão Martins Pinheiro

Mestrado Integrado em Engenharia Mecânica

Especialização em Automação

Supervisor at FEUP: Joaquim Gabriel Magalhães Mendes

Supervisor at Siemens: Mohamed Fouad Ahmed Beltagy

September 21, 2020

Motorcycle modeling for eCVT-in-the-loop real-time hybrid testing

Simão Martins Pinheiro

Mestrado Integrado em Engenharia Mecânica

Especialização em Automação

September 21, 2020

Abstract

Recently the number of innovative techniques for the automotive industry which combine numerical simulations with physical testing have been increasing. This combination, called hybrid testing, mix both the efficiency of numerical simulations with realism of physical testing.

This dissertation, following this line of development, is centered around an Aprilia Mana 850 GT motorcycle's transmission (eCVT) test bench and how to perform innovative hybrid testing solutions in this physical component. The final goal of the project where this dissertation is inserted is to reach System-in-the-loop stage, or more specifically the eCVT-in-the-loop one, with a model based system testing approach, combining real testing with numerical simulations and models. As a consequence, this work main focus is, with a real-time platform constructed and connected to the eCVT physical system, the creation of different motorcycle subsystems' models to be integrated in the testing loop. To fulfill this need a motorcycle engine model and braking system model using *Simcenter Amesim* software and a multibody model resorting *Simcenter 3D Motion* software were built. A step-by-step construction explanation is made throughout the document.

Models' results and behavior were compared with literature given the fact they are characterized by being behavioral models and not supported on experimental data.

Keywords: Motorcycle modeling, eCVT-in-the-loop, System-in-the-loop, Engine modeling, Multibody modeling, Braking system modeling.

Resumo

Recentemente, o número de técnicas inovadoras no ramo da industria automóvel que combinam simulações numéricas com testes físicos e reais têm vindo a aumentar. Esta combinação, denominada de *hybrid testing*, mistura a eficiência das simulações numéricas com o realismo dos testes físicos.

Esta dissertação de mestrado, seguindo esta linha de desenvolvimento, está centrada em volta de uma banca de testes que possui uma transmissão (eCVT) de uma motocicleta da marca Aprilia e modelo Mana 850 GT, com o objetivo de originar soluções inovadoras no que toca à execução de testes híbridos neste componente físico. O objetivo final do projeto, onde esta dissertação de mestrado se encontra inserida, é alcançar o estágio de *System-in-the-loop*, ou mais especificamente o de *eCVT-in-the-loop*, através de uma abordagem *model based*, que combina testes reais com simulações numéricas e modelos. Como consequência, o foco deste trabalho, com a criação de uma plataforma de tempo real e conectada com o sistema físico eCVT, passa pelo desenvolvimento de diferentes modelos de subsistemas da motocicleta com o intuíto de serem integrados no loop de teste. Para corresponder a esta necessidade, foram criados um modelo do motor da motocicleta e do sistema de travagem recorendo ao software *Simcenter Amesim* e um modelo *multibody* recorrendo ao software *Simcenter 3D Motion*. Uma explicação passo a passo da construção dos modelos referidos é feita ao longo do documento.

Os resultados dos modelos construídos foram comparados com literatura existente, uma vez que estes modelos são caraterizados pelo seu comportamento ao invés de serem suportados e comparados com dados experimentais.

Acknowledgements

Firstly I would like to thank Siemens Digital Industries Software for the internship opportunity and to allow me to develop a master's dissertation there.

I would also like to express my sincere gratitude to my supervisors Joaquim Mendes and Mohamed Beltagy for all the patience, guidance and support given throughout these 7 months and to Roland Pastorino for all the mechanical engineering insight and knowledge shared which allowed me to continuously progress and improve my work.

I would also like to extend my sincere thank to Marco Grottoli to guide me through the motorcycle multibody model construction and to facilitate my first steps with *Simcenter 3D Motion*.

Furthermore, I would also like to thank Egor Mozhenkov for the initial push and ideas shared to build the engine model with *Simcenter Amesim* software.

On a more personal note I would like to thank my family, more specifically my parents and brother, for giving me all the love and education i need.

Simão Martins Pinheiro

“Speed has never killed anyone,
suddenly becoming stationary... That’s what gets you.”

Jeremy Clarkson

Contents

List of Figures

List of Tables

Abbreviations

ABS	Anti-lock Braking System
AFR	Air/Fuel Ratio
ATDC	After Top Dead Center
BDC	Bottom Dead Center
BDF	Backward Differentiation Formula
BTDC	Before Top Dead Center
CFD	Computational Fluid Dynamics
CI	Compression Ignited
COG	Center Of Gravity
CT	Continuous Time co-simulation
DAE	Differential Algebraic Equations
DASSL	Differential/Algebraic System Solver
DE	Discrete Event co-simulation
DEEM	Discrete Event Engine Model
DOF	Degrees Of Freedom
ECU	Engine Control Unit
eCVT	electronically controlled Continuous Variable Transmission
ESP	Electronic Stability Program
EVC	Exhaust Valve Closing
EVO	Exhaust Valve Opening
FMEP	Friction Mean Effective Pressure
FMI	Functional Mock-up Interface standard
FMU	Functional Mock-up Unit
FOH	First-Order Hold
HCD	Hydraulic Component Design library
HiL	Hardware-in-the-Loop
HLA	High Level Architecture standard
HR	Hydraulic Resistance library
HYD	Hydraulic library
IC	Internal Combustion
IFP	French Institute of Petroleum
IVC	Intake Valve Closing
IVO	Intake Valve Opening
LHV	Lower Heating Value
MAF	Mass Air Flow
MAP	Manifold Absolute Pressure
MBD	Model-Based Development
MBST	Model-Based System Testing
MiL	Model-in-the-Loop

MVEM	Mean Value Engine Model
PECE	Predict, Evaluate, Correct, Evaluate
RPM	Revolutions Per Minute
RT	Real-time
RTOS	Real-Time Operating System
SI	Spark-plug Ignited
SiL	Software-in-the-Loop
SM	Surrogate Model
SOH	Second-Order Hold
SU	Simulation Unit
TDC	Top Dead Center
TPS	Throttle Position Sensor
TRIL	Test-Rig-in-the-Loop
UUT	Unit Under Test
WOT	Wide Open Throttle
XiL	X-in-the-Loop
ZOH	Zero-Order Hold

Chapter 1

Introduction

Recently the number of innovative techniques for the automotive industry which combine numerical simulations with physical testing have been increasing. These kind of techniques allow to perform what is called hybrid testing, reflecting the fact that part of an automotive structure (e.g. powertrain or a braking system) is analytically modeled while the remainder is physically tested, or vice-versa. In other words, it combines the efficiency of numerical simulation with the realism of physical testing [1]. This new automotive industry paradigm has made profound changes in the automotive systems development stages and design: it has transitioned from a labor-intensive and trial and error approach to a model based development, where a bigger variety of tests can be made and without even creating any complete prototype that can turn out useless or wrong in future stages of development and thus reducing cost associated with exclusive physical testing.

1.1 Motivation

This dissertation, developed at Siemens Digital Industries Software (Belgium), gives continuity to the above mentioned approach pushing forward an already existent project in this company, whose main goal is to demonstrate the full potential of company's software available to costumers, but also show how they can be connected in a real-time co-simulation environment where one of the system units is a physical component, broadening horizons when it comes to perform even better automotive testing solutions.

The project is centered around a physical motorcycle transmission, more specifically an eCVT (electronically controlled continuous variable transmission), from an Aprilia Mana 850 GT, represented in Fig. 3.1. The final purpose is to perform real-time co-simulations between different types of Siemens' software models of motorcycle's specific parts and the eCVT physical component, demonstrating one of a large group of testing solutions which can be configured with these kind of tools.

Prior work to this dissertation was related to the construction and upgrade of an eCVT test bench and designing the real-time platform in order to make possible the integration of future numerical models built to perform real-time co-simulations. The eCVT test bench can be seen in Fig. 1.1. The project has been developed by Siemens Digital Industries Software's testing division, more specifically the Model Based System Testing (MBST) team, responsible to implement innovative ideas related to hybrid testing solutions by developing numerical simulations and models, code, test benches and simulators to guide costumers into the future of automotive testing.

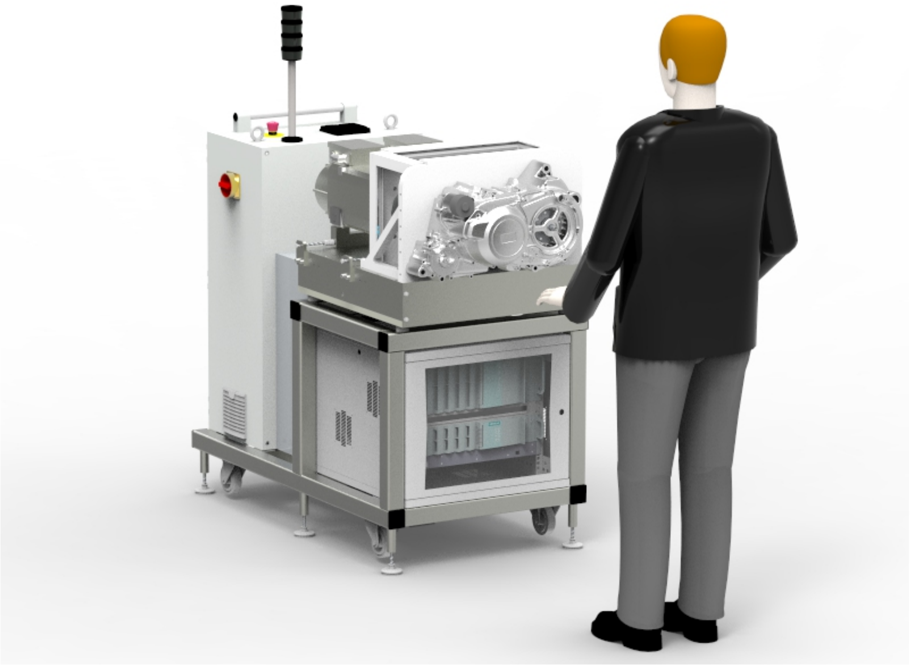


Figure 1.1: Siemens eCVT test bench

1.2 Objectives

To achieve project continuity the objectives of this dissertation were centered around developing motorcycle subsystems simulations resorting different Siemens software. Software packages expected to be used were *Simcenter Amesim* and *Simcenter 3D Motion*. In short, it was expected to complete the following tasks:

- **Engine model** — development of a motorcycle engine model resorting *Simcenter Amesim* to replicate engine output torque and rotary velocity.
- **Multibody model** — development of a motorcycle multibody model resorting *Simcenter 3D Motion* to replicate motorcycle body dynamics.
- **Braking system model** — development of a motorcycle front and rear braking system resorting *Simcenter Amesim* to replicate front and rear braking torques.

The way these models relate (with each other and with the real eCVT test bench) was needed to be previously defined as it influences the way each one of them is designed and built. This necessity corresponds to an important and previous task to the three mentioned before, so that in the future, is possible to perform real-time co-simulations. More insight about these relations, the test bench composition and real-time platform created in previous work is given in chapter 3.

1.3 Document structure

This dissertation is composed by eight chapters: The current chapter consists of an introduction where the context, motivation and objectives are presented. Chapter 2 presents the necessary theoretical concepts derived from the Model Based Development approach like Model Based System Testing or XiL technology. An introduction to real-time systems and simulations is also given although it won't be used for the current work. Finally a broad picture of the different types of simulations built along the course of this dissertation is also presented as well as the used software packages capabilities and utilities. Chapter 3 deconstructs the eCVT test bench and the future connections between motorcycle models and the physical eCVT (main goal of the aimed System-in-the-loop real-time hybrid testing). Chapter 4 gives a step-by-step engine model construction. Chapter 5 shows a step-by-step multibody model construction. Chapter 6 demonstrates a step-by-step braking system model construction. Chapter 7 presents the results of each model and analyzes their behavior comparing to existent literature. Chapter 8 exhibits a final insight on this work. Future work is also proposed.

Chapter 2

Literature Review

In this chapter it will be covered general but important topics about automotive testing and simulations to give some context to the reader about the project, finishing with more detailed subjects regarding motorcycle modeling which are the main purpose of this dissertation.

Beginning introducing vehicles development approach with Model Based Development (section 2.1), Model Based System Testing (section 2.2) and XiL Technology (section 2.3), we jump into defining real-time systems and simulations (section 2.4), either running in a stand alone manner or coupled with each other (co-simulations). Systems simplification that comes along with the necessity of running models in real time is also briefly explained. To conclude this chapter, it is given a brief literature review on modeling different parts of a motorcycle (section 2.5) resorting different modeling tools as well. Besides the motorcycle dynamics, the parts covered were the motorcycle's engine and braking system.

2.1 Model based development

Back in 2012 there were already statements informing that only within a few years the share of software controlled innovations in the automotive industry had increased from 20 to 80 percent, and it seem to keep growing. In fact, forecasts at the time claimed that software will determine more than 90% of the functionality of automotive systems within the next decade. This continuous growth has become a reality and strengthen and solidified a development process dated back to those past days, founded on model based technologies which have many advantages for automotive developments [2]. This process is supported by software tools that can be used to rapidly visualize the control algorithms and refine the requirements, becoming the first task of design and implementation.

Before this developments, the main and most traditional approach was characterized by being centered around a labor-intensive and trial and error process of design, implementation and testing, which is much more resource and time consuming [3]. Given this characteristics, a failing attempt to build something can become a demanding task as there is a need to redesign if a prototype must be abandoned [4]. In order to overcome these disadvantages, the model-based technologies allowed the development of high-level and more complex models that can be suited for simulation in the first steps of the development process. As automotive development is a very interdisciplinary business, the existence of graphical models and their simulations allows common functional understanding between engineers, early in the design phase [2].

2.2 Model based system testing

As stated previously, the development methodology has evolved from test-centric to a model-based development (MBD) approach, however, its practical realization relies on the quality of the models. So, to enhance the quality and the approach productivity there is a framework of engineering solutions called Model-Based system testing (MBST) which optimally balances the combined used of test and simulation [5]. This way test data can be used to validate models but also for more complex and realistic interactions, combining models with experimental data into hybrid testing approaches [6]. In few words the MBST is defined as the discipline that brings physical testing and simulation models together with the aim to study, identify, validate and improve the behavior of multiphysical and mechatronic systems. It deals with multiple types of simulation models: structural test and/or 3D, but also with multibody and 0D/1D/2D/3D multiphysical simulation models, depending on the possible computational complexity [7]. As a matter of fact, state-of-the-art regarding simulation solutions state that it is already possible to accurately simulate the multi-physical nature of systems by incorporating all relevant physical phenomena (mechanics, electrics, hydraulic, pneumatic, etc.) in one solver [5]. MBST can be divided into three main categories, as can be seen in Fig. 2.1:

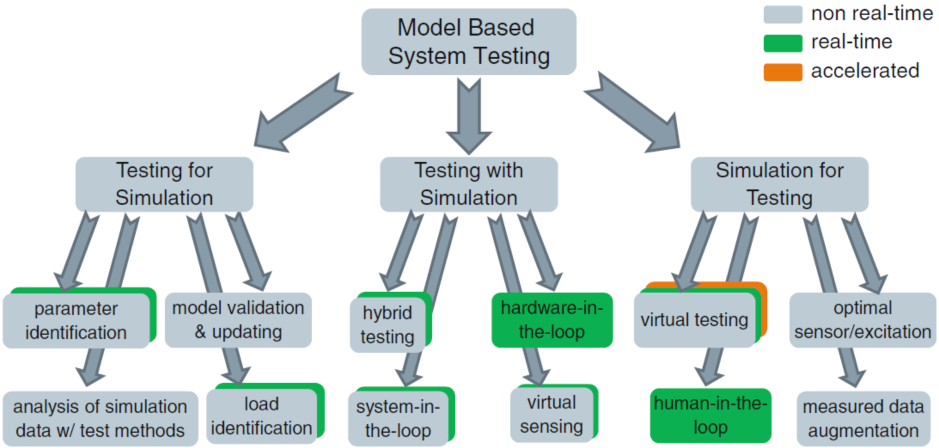


Figure 2.1: Model Based System Testing application tree [7].

- **Testing for Simulation** — or more specifically, test data, remains crucial to offer input to critical model parameters and to build, validate, improve and drive numerical models in order to provide the level of accuracy and completeness that a sole use of simulation models cannot give in many applications [6]. To gather and collect the test data it is required data acquisition solutions from a wide variety of sensors (for example, flow sensors, anemometers, accelerometers, inertial motion units, microphones, thermocouples, among others) [5].

- **Simulation for Testing** — is the modality allowing faster and improved testing processes by mirroring the test conditions through the use of simulation resources. It is extremely important to have the optimal test conditions in the design phase like optimal sensor location (optimal observability), excitation placement (controllability) and meaningful sensor results (correct calibration, wiring, configuration, etc.). Simulation models are used to accomplish these requirements by checking, validating and optimizing the complete test process before performing the actual test. Another important advantage of this modality is the cost reduction by the use of virtual testing on expensive or safety-critical or hard accessing systems [5];

- **Testing with Simulation** — unlike the first two cases where simulation and testing are decoupled, in this modality this two actions are tightly coupled [7]. It reduces the development time and cost and decreases the risk of discovering flaws when introducing a physical component in a system late in the development phase. Relative to the previous two cases, the models needed are more advanced and should met special criteria - they should replicate with accuracy the physical behavior of the parts intended to simulate and for some cases must be suited for real-time applications. In other words, these models should be obtained by reduced sized models and other order reduction techniques to obtain very low computational times [5].

2.3 XiL technology

XiL (X-in-the-Loop) technology began to be studied in 1996 being considered later as a philosophy for vehicle research and development [8]. The first proposed concept appeared in Germany and characterized X-in-the-Loop as the following: an integrated in-loop method where X refers to the Unit-Under-Test (UUT), which can be a model (Model-in-the-Loop, MiL), software (Software-in-the-Loop, SiL) and hardware (Hardware-in-the-Loop, HiL) [9]. HiL is still the most popular approach for simulation-focused testing in the automotive field but MiL and SiL have been more frequently used [10]. Variants from XiL technology have been also emerging; one example of that is the test-rig-in-the-loop (TRIL) concept where the aim is to obtain real-time integration of two or more test rigs, either locally (distance between over 1km) or remotely (between different geographical locations with the help of a global server) [11]. This technology is deeply related to the previously discussed topics, as it is used in the MBD approach and in the modality "Testing with Simulation" to perform proper real-time simulations. Normally, three validation stages take part of the MBD tool-chain, needed to accomplish this type of simulations:

First stage is MiL, where the goal is to build some simulation models to verify requirements and algorithms as well as generating reference test outputs for the following validation stages. There is no need for special hardware use as everything is made offline combining multi-physical with control models. It is also possible to use co-simulation approaches taking advantage of different software tools capabilities (for example: performing control engineering in MATLAB/Simulink and system engineering in Simcenter Amesim).

SiL makes the **second stage** granting first software validation and adaptation for real-time computation. It replaces the full simulation model into an executable code running on a standard computer in a fixed-point manner [6].

The last and **third stage** is the XiL one, where X refers to a real system, which can differ depending on the different types of testing [5]:

- **Hardware-in-the-loop testing** — only electronic controllers, for instance, engine control units or transmission control units, are physically present on the test bench which in its turn are connected to a real-time platform simulating the rest of the system.

- **System-in-the-loop testing** — some components or a (sub)system, for instance, a vehicle driveline or a steering actuator, are physically present on the test bench which in turn are connected to a real-time platform as well.

- **Human-in-the-loop testing** — in this special type of testing a human user can interact with a simulation of the machine. Suitable examples are driving or fligh simulators.

Figure 2.2 demonstrates an example of an automotive electric powertrain testing using MBD tool-chain approach and XiL technology [6].

2.4 Real-time systems and simulations

2.4.1 Definition for real-time

Although it may seem there is no need to define real-time systems or simulations because of its obvious and intuitive meaning, the term is widely used in different contexts and can sometimes generate misconcepts. Some frequently cited misconceptions, for example, state that real-time is synonymous with "fast" systems. Thus, it is important and necessary to fully comprehend when a system or simulation is real-time or not.

In order to do that, lets first define some central terms about this topic. As is common sense, a system is an organized assembly of components mapping a set of inputs into a set of outputs with a specific purpose. The time between the presentation of a set of inputs and the response outputs (called the system response time) must be initially set and satisfied if the system is performing in real-time. Failing to accomplish this time constraint there is a severe risk of consequences,

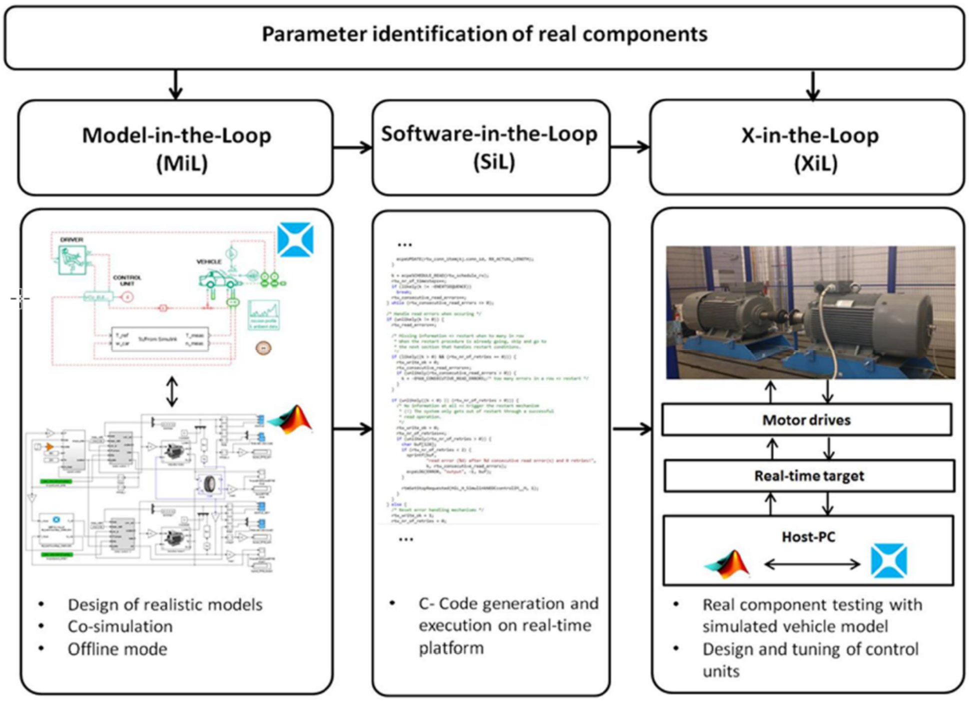

Parameter identification of real components
Model-in-the-Loop (MiL)
Software-in-the-Loop (SiL)
X-in-the-Loop (XiL)
Motor drives
Real-time target
Host-PC
• Design of realistic models
• Co-simulation
• Offline mode
• C- Code generation and execution on real-time platform
• Real component testing with simulated vehicle model
• Design and tuning of control units

Figure 2.2: Automotive electric powertrain testing using MBD approach and XiL technology [6].

including system failure. In other words it will cause a system inability to perform according to system specifications. In short, and to reject the misconception that each real-time system task has to process data at once or instantaneously, it simply has response times that are constrained appropriately and at some degree are effectively satisfied. Even a system with response time of days or even weeks, for instance, a bimonthly payroll run, could be defined as a real-time system. In an extreme case, if ultimately it was timeless, every system could become a real-time system.

For engineers the ultimate goal is to find cost-effective ways to process everything as slowly as possible and repeat task as rarely as possible always having in mind that the system response time depends on which application they have at hands (an aircraft, for example, must process accelerometers data in the order of microseconds) [12]. It is also significant to take into account the degree of importance in accomplish all the response time constraints to the system good performance. It depends massively on the application scenario and there is a full scale of real-time systems that go from soft to hard, presented in Tab. 2.1.

Briefly, real-time systems can be characterized by five different requirements. A system must be **deterministic** as it must have unambiguously output information for each input information and unambiguously amount of next states with the response time for all output information determined (determinism in time). Besides, should have **promptness** by processing tasks just in time with more or less hardware according to performances needed (depending on system levels and the application scenario as shown in Tab. 2.1. Another important characteristic is **simultaneity** in the sense that different tasks must be running at the same time, like controlling or measuring tasks. However, to reduce costs it is common for systems to run quasi-simultaneously to reduce

Table 2.1: Real-time levels [12].

Level	**Description**
hard RT	Has to ensure that a task is done until a deadline. Missing the deadline causes a system failure, e.g., ABS.
firm RT	Tolerates infrequent missing of deadline while processing. However, exceeding the deadline criteria reduces the quality of service. The outcome after missing a deadline can't be used, e.g., multimedia.
soft RT	For a certain threshold it is allowed to miss a deadline. The outcome after missing a deadline is usable, but reduces the quality of service, e.g., air conditioner.

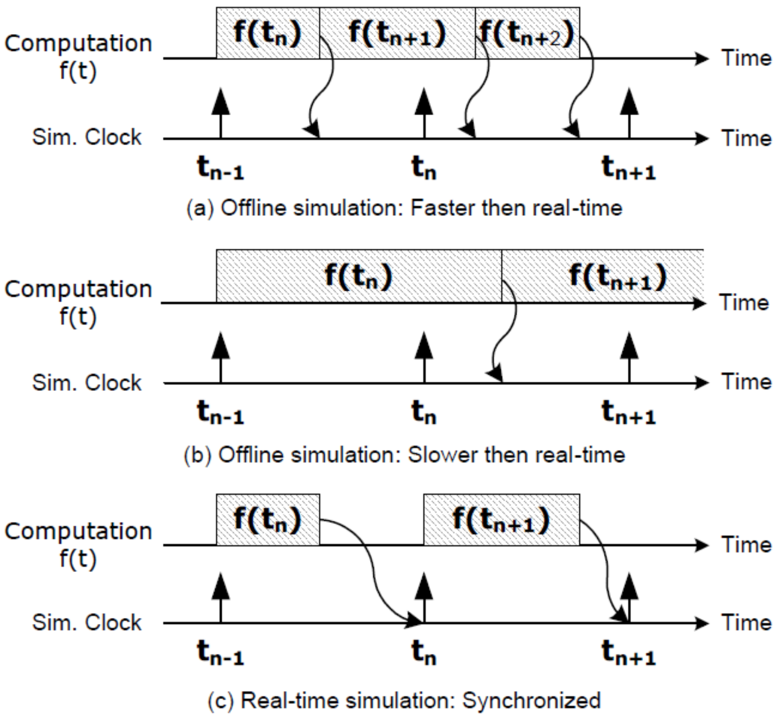
efforts, meaning that one or multiple processors handle multiple task instead of the existence of one processor for each task. **Just-in-time response to spontaneous events** can also characterize a real-time system, resorting to polling (a cyclic request of all input changes) and to interrupts, when handling external events or significant condition changes. Last but not least, a system must be **reliable** just to ensure safe operation even if some hardware fails or is damaged. In order to do that, hardware components are designed with redundancy [13].

2.4.2 Fixed time step solver and real-time simulations

The current master thesis will only focus on one simulation type, where time is discrete and moves forward in steps of equal duration, also known as fixed time-step simulation. There is also the possibility to conduct other techniques with variable time-steps which are not suitable for real-time applications and as a result are not important to deepen. For each time step, for the first described simulations, there is a set of mathematical functions and equations to solve variables or system states with the input from variables and states from the preceding time-step. The time-step during offline simulations can be longer or shorter than the amount of real time required for performing real-time time-steps. Accelerated simulations are referred if the offline time-step is shorter than the real time one, as can be seen in Fig. 2.3 (a), while in Fig. 2.3 (b) the offline computing time is longer than in real time.

In offline simulations the main objective is to get results as fast as possible (which is dependent on the computation power and system's model complexity), but the moment at which the results become available doesn't really matter.

In contrast with offline simulations, real-time simulations validation depends if the simulator used accurately obtain internal variables and outputs within the same time length as its physical counterpart would. In more precise terms, the time length should be even a little shorter to allow the real-time simulation to perform all operations like including driving inputs and outputs to and from externally connected devices, before the next time-step (see Fig. 2.3 (c)). In this type of simulation, even if all computations are done before the time-step is concluded, there is no



(a) Offline simulation: Faster then real-time

(b) Offline simulation: Slower then real-time

(c) Real-time simulation: Synchronized

Figure 2.3: Real-time simulations requisites [14].

possibility (like in accelerated offline simulations) to compute the equations in the next time-step. So this waste of time where no operation is performed and the simulator waits for the next time-step is called **idle-time**. On the opposite case, **overrun** happens if all the calculations haven't been made within the required time-step. The real-time simulation in this case is considered erroneous. Another important topic is the proper determination of the **time-step duration**. The correct minimum time-step will allow the simulator to achieve real-time without overruns and can be dictated by the frequency of the highest transient of interest. In parallel to this information there is a rule of thumb which states that the simulation time-step should be smaller than 5% to 10% of the smallest time constant of the system. Also, in the state-of-the-art there are already time-steps as low as 10 microseconds. In Fig. 2.4 it is outlined the simulation's time-step for different applications. It is worth to notice that mechanical systems and vehicles belong to the category of low speed and low power computation systems [14].

2.4.3 Co-simulations

In the automotive industry there are many specific simulation tools to replicate different parts and subsystems of a vehicle, made specially for different engineering areas and with specific types of solvers. Given this panorama, a big problem consists in generating a heterogeneous simulation that can simulate all the interactions between different vehicle subsystems and the environment, coupling all in one the different specific domain simulations [15]. To keep benefiting

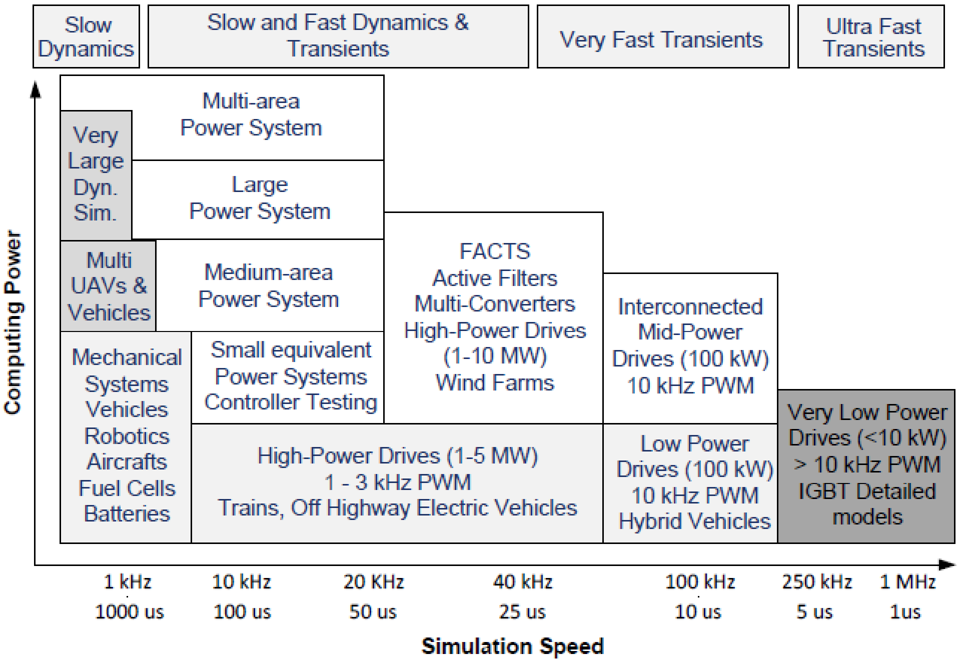

Slow Dynamics
Slow and Fast Dynamics & Transients
Very Fast Transients
Ultra Fast Transients
Computing Power
Multi-area Power System
Very Large Dyn. Sim.
Large Power System
Multi UAVs & Vehicles
Medium-area Power System
FACTS
Active Filters
Multi-Converters
High-Power Drives (1-10 MW)
Wind Farms
Interconnected Mid-Power Drives (100 kW) 10 kHz PWM
Mechanical Systems Vehicles Robotics Aircrafts Fuel Cells Batteries
Small equivalent Power Systems Controller Testing
High-Power Drives (1-5 MW) 1 - 3 kHz PWM Trains, Off Highway Electric Vehicles
Low Power Drives (100 kW) 10 kHz PWM Hybrid Vehicles
Very Low Power Drives (<10 kW) > 10 kHz PWM IGBT Detailed models
1 kHz 1000 us
10 kHz 100 us
20 KHz 50 us
40 kHz 25 us
100 kHz 10 us
250 kHz 5 us
1 MHz 1us
Simulation Speed

Figure 2.4: Time-step simulation by application [14].

from simulation results and analysis, there are two ways to simulate the interactions between subsystems mentioned above: the entire systems is modeled with a single simulation tool, a less flexible approach, or the respective subsystems' simulations can be coupled in a so-called **co-simulation**, where each one of them is defined as an independent black box (called simulation unit, SU) capable of exhibiting behavior, consuming inputs and producing outputs. To exhibit behavior, or in other words, to compute the behavior trace[1], a simulation unit must have the following requirements:

A **model**, created by the modeler based on the characteristic of the system under study. A **solver**, the program algorithm tool used to compute the behavior trace of the dynamical system as close as possible from reality. Over time there is also a necessity to perform some extrapolation/interpolations to get an **input approximation** of the model, used by the solver [16, 17]. This happens in cases when, for instance, two subsystems are coupled and interdependent, and therefore at least one input has to be extrapolated to solve the coupled overall system. Other case could happen when at the beginning of a time step there is no input available for the SU so there is a need to get an approximation of its value [15]. Finally there is also a need for the notion of **input and output reactivity** important to establish which inputs does the simulation unit will receive from other co-simulation components.

From a bigger picture, in order to run a co-simulation one needs a co-simulation **scenario** and **orchestrator**:

[1] set of trajectories followed by the state of the dynamical system when inputs are provided.

- **Orchestrator** — As the black boxes are independent from each other an orchestrator is responsible for producing the co-simulation results and for initializing all the SUs with the appropriate values, setting/getting their inputs/outputs, and coordinating their each own simulated time progress.

- **Scenario** — points to one or various SUs and describes for each one of them how the inputs and outputs of their models are related and coupled.

Current state-of-the-art distinguishes two main orchestrators and co-simulation paradigms: discrete event (DE) and continuous time (CT).

A **discrete event** co-simulation operates the communication between SUs resorting events as opposed to signals meaning that SUs's outputs can be absent at times where no event is produced. The two main characteristics of this paradigm is reactivity - the capacity to process an event at the moment it occurs, and transiency - an event can cause other events to occur instantaneously.

A **continuous time** co-simulation, in theory, allows SUs to exchange their data continuously. However, in practice, as this exchange is not possible, there is a group of techniques applied to compensate this limitation, which are mentioned below.

In a co-simulation context, the step size of a SU is typically called micro time step. As each SU's micro step size is independent, a communication step size H between SU's, also called macro step size, has to be defined to perform a continuous time co-simulation. This is one of the most critical parameters that evolve this type of simulation: its value normally relates to a trade-off between the overall simulation time and accuracy required. The rule of thumb regarding the mentioned trade-off would be that smaller macro time steps lead to longer simulation times but more accurate simulations [15, 16, 17]. The macro step size H is normally fixed in practical applications, but further improvements are expected from adaptive macro step sizes that may handle better the various changing dynamics of models [18].

Another important issue is the communication approach, the order which SUs are given inputs and instructed to compute the next interval:

- **Sequential or Gauss-Seidel communication approach** — orchestrator instructs one simulation unit at a time to produce outputs until the next simulation macro time step. The most recent results are then used by the orchestrator to instruct the next unit. With this approach a new problem that also influences the overall results arises - which system is simulated first? One of its advantages is that only one of the inputs of a two coupled system needs to be extrapolated resulting in less coupling errors but bigger computational times.

- **Parallel or Jacobi communication approach** — oschestrator instructs all the simulation units at once setting their inputs at the end of each time interval. Opposite to Gauss-Seidel approach all the inputs have to be extrapolated resulting in bigger coupling errors, as more extrapolations are performed, but less computational times.

Typical co-simulations include algebraic loops either in cases when inputs of a SU depend on its outputs or when implicit numerical solvers are used influencing state variables as well. For the second case there are three different ways an orchestrator can handle the situation:

- **Fully implicit iteration or strong coupling** — for those involving state variables, the same co-simulation step has to be repeated until convergence or when the introduced error is below a predefined threshold. It makes use of what is called *rollback*.
- **Semi-implicit iteration** — a fixed number of steps or iterations are performed.
- **Non-iterative or weak coupling** — no iteration is performed.

Figure 2.5 represents the different mentioned communication approaches and ways of dealing with algebraic loops.

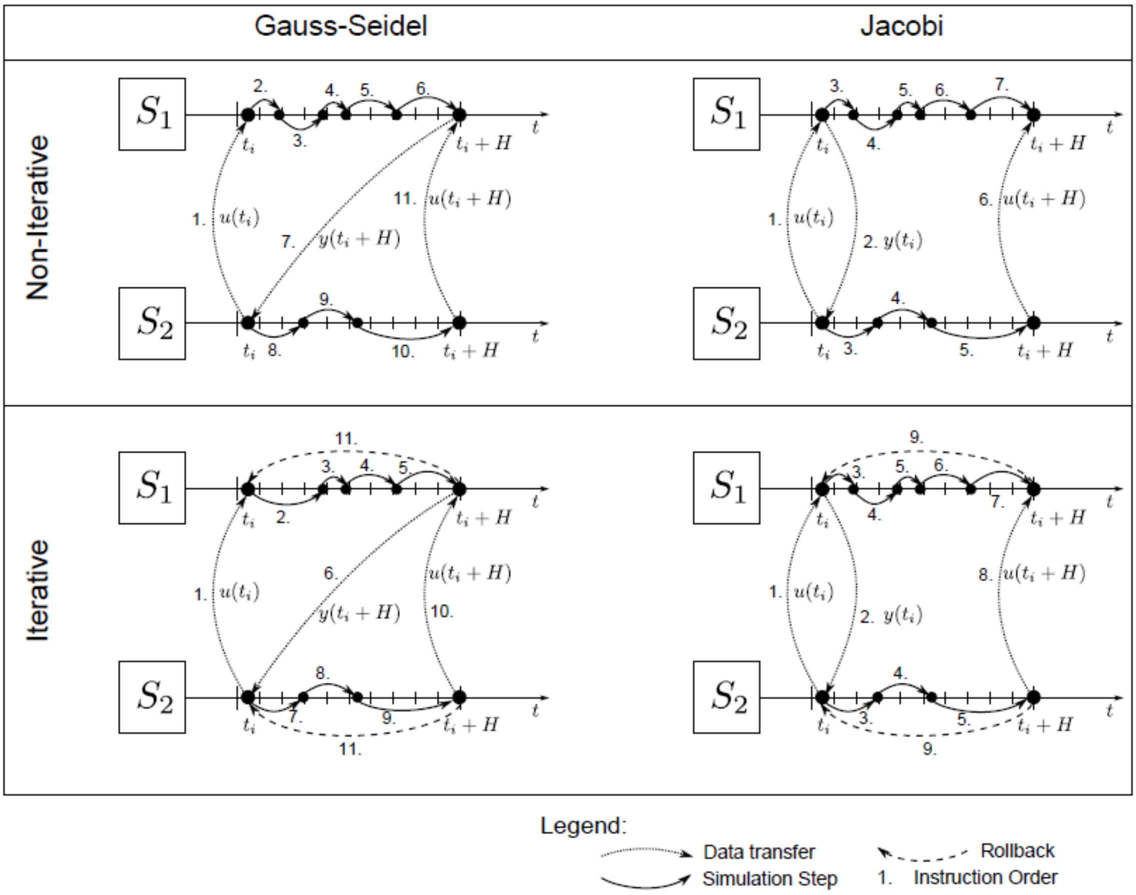


Figure 2.5: Communication approaches and ways of dealing with algebraic loops [16].

It is important to state that CT and DE co-simulation are not two independent branches and in fact, with increasing research and development, hybrid co-simulation, where a mix of those two paradigms are taken into account, are emerging. Figure 2.6 resumes the co-simulation topic in a schematic diagram.

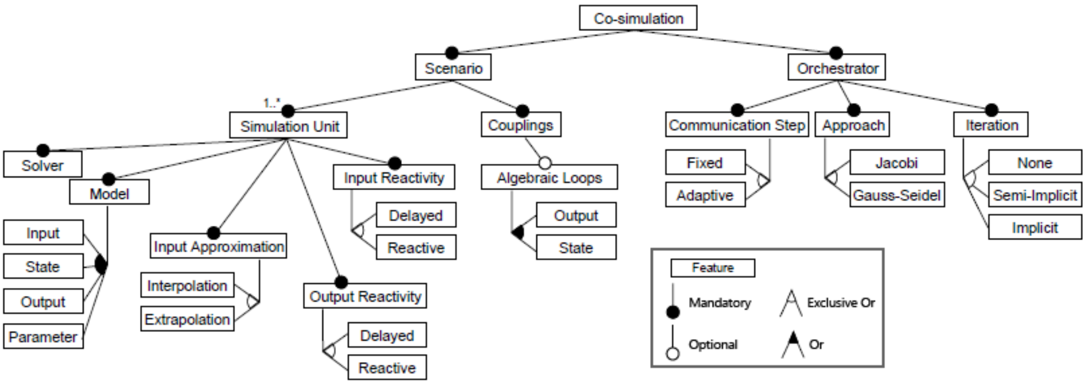


Figure 2.6: Co-simulation overview [16].

Standards were created to define the communication interface and protocol between the SUs and orchestrator. Some of the well-known standards are the Functional Mock-up Interface (FMI), suited for CT co-simulations, and the High Level Architecture (HLA), suited for DE co-simulations [15, 16, 17].

2.4.4 Real-time Co-simulations

When in a co-simulation one of the present subsystems is a real-time system, the simulation is called real-time co-simulation, bringing increased difficulties to guarantee hard real-time conditions unless a correct coupling between subsystems is used. Additional problems will be raised besides the ones already existing in a classical co-simulation. In this kind of simulation, time accuracy comes into place as the non real-time systems have to be synchronized with the hard real-time systems, operating with clock time. Another important problem, as real hardware enters the overall simulation, is the handling of noisy coupling signals coming from sensors' measurements. Table 2.2 shows the specific characteristic of a real-time co-simulation [15].

2.4.5 System identification and simplification

In some cases modeled systems have a very poor performance, taking too much time to solve and calculate simulation results. In cases where the model should be real-time compliant or the computational costs need to be reduced, simulation engineers must identify the system's critical points and simplify the model according to them. Depending on the outcome pretended, the system analysis can be made in the time domain, by the values of quantity *h(t)* as a function of time *t*, or the frequency domain, where the process is specified by giving its amplitude *H(f)* as a function of frequency *f*. However, experts tend to pick frequency-domain because analyzing results in time domain can be extremely difficult and takes longer CPU-times [19]. Linearizing the dynamic system around specific operating points it is also the *modus operandi* when the purpose is to analyze and take conclusions on system dynamic properties. The linearized system is normally represented either with its transfer function *H(s)* in the Laplace formalism, or with state-space

Table 2.2: Real-time Co-simulation main characteristics

Characteristic	**Description**
Extrapolation techniques	Typical extrapolation techniques used are zero order hold (ZOH), first-order hold (FOH) and second-order hold (SOH) as long as coupling signals are not corrupted by noise.
Error correction	Errors introduced by extrapolations can not only affect the true solution but in some worst case scenarios may be responsible for an unstable co-simulation. Because of that, it is needed for the coupling element to operate some error correction techniques.
Dead-times compensation	Times where data is being exchange must be minimized to guarantee hard real-time conditions.
Synchronization	The coupling element must synchronize systems that satisfy real-time conditions with ones who don't.
Macro time step	The macro time step size is strongly related to the real-time systems. It must be smaller or equal than the fastest update rate of the real-time systems.
Communication approach	Jacobi approach is preferred with faster execution times. However, if Gauss-Seidel approach doesn't violate the hard real-time conditions, it can be used with a smaller coupling error advantage.
Iterations	Fully implicit iterations are discarded because the run time behavior becomes unpredictable. Semi-implicit iterations can be used only if the number of iterations made have an execution time smaller than the macro time step.

representation [20]. Resorting the above mentioned approaches, the dynamic system behavior is studied through:

- **Eigenvalues** — representing the system natural modes. Prediction of possible system resonances at specific frequencies made possible, as well as the associated oscillations damping ratio.

- **Modal shapes** — representing the space distribution for a natural mode (i.e. frequency and damping ratio) all along the system.

- **Transfer Functions** — representing the frequency response in gain G (dB) and phase φ (degrees) of an observer variable (output) linked to the excitation of a control variable (input).

- **Root Locus** — plot (real part/imaginary part), representing the trajectories in frequency and damping ratio of the natural modes due to some parameters changes.

Bode, Nichols and Nyquist plots are also used.

2.4.5.1 Model reduction and speed-up

Computation time taken to simulate models depends on various factors, namely the number and type of equations being solved, the number of discontinuities, speed at which variables change in the model (very high natural frequencies can explain why the simulation is slow), hardware and network infrastructure including computer speed, among others. To decrease the computational time some standard procedures can be done:

- **Keep the model in a specific frequency bandwidth (Hz) of observation and interest** — For example, if the dynamic system in study has a [0-100 Hz] bandwidth of interest, it is pointless to take into account natural modes up to 5000 Hz since they may slow down the simulation and no useful information will come from the result with [100-5000 Hz] bandwidth.

- **Avoid discontinuities** — In case there are too many discontinuities, try to avoid the ones which are not essential.

- **Reducing state variables number** — The number of state variables has a direct influence on the computational time.

- **Different running simulation approach** — Instead of running the simulation on a single computer, a good way of reducing computational effort would be to distribute the runs over several processors of the same or different machines.

- **Discrete partitioning** — Dividing a complex model into smaller and parallel sub-models that run as a sort of co-simulation will speed up the time.

There is also the possibility that the model needs to be big, complex and with very high frequency eigenvalues resulting in an impossibility to slow down the simulation [19].

2.4.5.2 Surrogate modeling

In cases where no more system simplifications are achievable or intellectual property wants to be preserved, but there is still a need to reduce computational times, surrogate models are implemented. Surrogate models (SMs) are approximations that are fit to the available data and make a functional relationship between input variables and the output quantity of interest. In other words, unlike the speed-up techniques mentioned before, intended to simplify the model but keeping the physic laws untouched, the SMs are an alternative way to represent the system in a numerical approach [21]. An extensive amount of surrogate modeling methods are presented in literature. One interesting and more recent approach is based on artificial neural networks [22].

2.5 Motorcycle simulations

2.5.1 Multibody modeling

It is widely recognized that, as far as control systems design is concerned, dealing with motorcycle dynamics is more complex and subtle than with four-wheeled vehicles, once a complete analytical model is not directly available due to its high sensitivity to parameters' variations. While for cars it is possible to implement control systems based on simplified dynamical models (e.g the quarter car model), unfortunately, the same doesn't apply for two-wheeled vehicles. The presence of a single axle doesn't allow model simplifications and tire-road interactions effects and tire modeling can never be ignored or omitted, being in fact the core of an effective motorcycle model and having a direct impact on both ride and handling properties of the vehicle. Other crucial difference between cars and motorcycles' dynamics is the driver impact: on 4 wheeled vehicles the driver impact is only considered as a measurable disturbance (i.e., the steering angle); on 2 wheeled vehicles the driver current position and mass distribution while driving are key variables for defining the motorcycle instantaneous center of gravity. However, for the sake of simplification, there are cases where the driver is considered attached to the motorcycle rear frame center of gravity and as consequence it is not taken into account this particularity [23, 24].

In the past, most of the work carried around motorcycle dynamics involved studies using theoretical models derived explicitly. However, more recently there has been an increasing use of computer assisted software made of multi-body formalisms, boosting the models' complexity [23, 24].

In multibody dynamics formulation all structural components of the system (motorcycle and driver body) are represented by several individual discrete bodies, each one of them with a maximum of six DOF, corresponding to three translational and three rotational components. All bodies have a mass and inertia properties, but cannot deform. Furthermore, they are connected through joints that represent the flexibility and damping in the system. Two different approaches can be applied according to the problem at hands:

- **Forward dynamics** — predicts the multibody system behavior and motion over a given period of time, as a consequence of applied forces and given initial conditions. Useful for the study of motorcycle motion as a consequence of engine and braking torque applied to the system.

- **Inverse dynamics** — Determines the driving forces that produces a specific motion in the multibody system and calculates the reactions forces present in each of the joints connecting the discrete bodies. For instance, with this approach it is possible to predict the steering torque applied by the driver to be able to perform a given curve with his motorcycle.

Initial motorcycle conditions or configuration, before the simulation starts, is solved as a static equilibrium problem depending on the weight and position of the bodies center of gravity and not on their inertia properties. This problem leads to a system of nonlinear equations solved iteratively [25].

For this master thesis the commercial software used to develop the motorcycle multibody model was the *Simcenter 3D Motion*, so further details are given about the equations and calculations to predict motorcycle motion. As for any mechanism modeled in *Simcenter 3D Motion*, a Newton-Euler formulation is used to determine the equations of motorcycle motion. To represent the position and orientation of a 3D body in space the motion solver uses a set of translational coordinates and the four Euler parameters.

$$q = [x, y, z, e_0, e_1, e_2, e_3] \tag{2.1}$$

The implementation of a joint, or in other words the removal of degrees of freedom to the system, is made by using algebraic equations which represent the joints kinematic formulations. Mathematically, this is represented as the following holonomic constraint:

$$\Phi(q,t) = 0 \tag{2.2}$$

The algebraic equations above mentioned are appended to the Newton-Euler equations of motion using a set of Lagrange multipliers, λ. External forces applied to the system results in a set of first-order state equations in the state variable, χ. With all this being said, the final form of the equations of motion including the first-order control state equations are written as:

$$M\dot{v} + \Phi_q^T \lambda - Q_a(q, \lambda, \chi) = 0 \tag{2.3}$$

$$\Phi(q,t) = 0 \tag{2.4}$$

$$\chi = g(q, v, \dot{v}, \lambda, \chi) \tag{2.5}$$

Where,

M = mass matrix

Q_a = vector of applied forces

Φ_q = Jacobian of algebraic constraints

v = velocities vector (velocities are the derivatives of equation 2.1)

Equations 2.3, 2.4 and 2.5 are an index-3 set of differential algebraic equations (DAE) and special techniques are used to integrate them over time. Two integration methods can be selected in the software: BDF (Backward Differentiation Formula) - implicit integrator based on DASSL; PECE (Predict, Evaluate, Correct, Evaluate) - an explicit integration Adams-Bashford-Moulton predictor corrector method [26]. Static equilibrium analysis of this set of motion equations does not require time integration, but instead an algebraic solution. Further, in an equilibrium analysis all system velocities are assumed to be zero. The result is the following set of algebraic equations of motion:

$$\Phi_q^T \lambda - Q_a(q, \lambda, \chi) = 0 \tag{2.6}$$

$$\Phi(q, t) = 0 \tag{2.7}$$

$$\tag{2.8}$$

A modified Newton's method is employed to solve this set of equations [27].

2.5.2 Engine Modeling

Modeling of internal combustion (IC) engines is an engineering area that deals with several disciplines, like thermodynamics, fluid mechanics, turbulence, heat transfer, combustion, chemical reactions, mathematical analysis, numerical methods, among others [28]. The main goal of this type of models is to predict the performance of an engine under different configurations and conditions before proceeding to real engine tests [29]. Along the course of history, models with different levels of complexity, approximations and forms were created. The models always depend on the desired outcome and available information, explaining the amount of differences from model to model.

The most simple and straightforward approach for engine performance simulation is the use of lookup tables containing different engine variables, as, for example, engine torque and speed or fuel consumption characteristics. However, this kind of models require extensive experimental data and therefore the approach to create them is categorized as non predictive. The introduction of physical-based models with meaningful parameters, tend to reduce the necessity of experimental data and approximate them to a more predictive approach [30].

Two big types of engine models appear in literature:

- **Mean value engine models (MVEM)** — evaluates engine's performance based on mean values of the thermodynamic parameters, combining the overall effect of engine flow and combustion phenomena, neglecting the instantaneous variations during discrete engine cycles (time is the independent variable). Such models require little physical detail and because of that some parameters must be derived from experimental testing. As a result, MVEM models have less accuracy but require less computational time, being one of the most resorted types in literature [29, 30].

- **Discrete event engine models (DEEM)** — explicitly takes into account the reciprocating engine behavior where the crankshaft angle is the independent variable. Obviously requires more computation time due to larger model complexity [31].

Other important topic, as previously mentioned, is the different degrees of complexity and predictability shown by the engine models. Literature review shows that, despite some ambiguous definitions found through some scientific articles, models can be classified in four big categories, namely zero-dimensional (0D) single zone model, 0D/1D single zone model, multi-zone models and multidimensional models. Being the last category the most complex and predictable and the first the most simple and least predictable [32]. The ambiguity comes precisely from the fact that one model can be composed from subsystems with different complexities and terminations. Normally, the whole complex engine system is composed by subsystems and submodels like the cylinder head, cylinders and crankshaft, air path and gas dynamics, injection, combustion, heat transfer in the combustion chamber or turbocharger.

For **0D/1D models** the engine is represented as network of pipes and volumes interconnecting the mentioned subsystems and submodels. These type of models can handle real-time simulations for transient engine operations and can be used to develop control strategies. On the negative side, they are not completely predictable and contain several empirical parameters. **0D models**, in particular, are characterized to have most of their properties averaged over the total volume and no spatial information available. MVEM models also fit this category [28, 32, 33].

Combustion models can be separated in two types: **single zone or non-predictive model** - the burned and unburned gases are mixed into a single volume, where the burn rate is imposed as a function of the crank angle. This rate will be followed regardless of the conditions in the cylinder; **multi zone or predictive model** - the combustion chamber is divide into two zones (burned and unburned gases) separated by a premixed turbulent flame. The burn rate is predicted from appropriate inputs (pressure, temperature, etc.). It is important to keep in mind that, a logical strategy would be to implement a non-predictive model when possible and to implement predictive models when necessary, due to their increased complexity and larger computational time [28, 32, 34].

If a higher level of detail is required to simulate engine flow and combustion, more complex and high predictable simulations have to be performed, like 3D computational fluid dynamics (CFD) models. It is normally employed when engine components geometry need to be optimized. However, it should be noticed that 3D simulations normally focus on one single engine component without interacting with other system components due to high computational costs. Nonetheless, some solutions are presented in literature where 0D/1D models simulate the whole engine except the component that will be studied in detail with a CFD model, with the purpose of generating the dynamic boundary conditions to a more complex simulation [28, 30, 32, 33].

For the context of this master thesis, the resorted tool to build engine models was the software *Simcenter Amesim*. Amesim environment is an open numerical platform for modeling and simulation of dynamic systems with multi-domain architectures, based on the Bond Graph and adopting an object-oriented programming language [34, 35]. It makes it possible to build a model aggregating or connecting simple components, in a both easy and intuitive way, resulting in a system layout similar to the physical one. The object oriented approach to physical systems is based on specific paradigms [23]:

- **Acausal modeling** — Equations of each submodel are written stating physical parameters rather than by writing computational algorithms, independently of the boundary conditions. The causality of the whole model is determined automatically by the *Amesim* environment at the moment which is assembled out all the elementary submodels.

- **Code transparency** — Easy to understand the content of the equations inside each submodel as they tightly match the way they are written.

- **Encapsulation** — The interaction between submodels is made through rigorously defined interfaces, called connectors. As long as two different submodels have compatible connectors they can bound together, regardless of their inner details.

- **Inheritance** — Submodels can have a hierarchical structure, where more complex submodels can be obtained by connecting simpler submodels with each other. *Amesim* software calls this feature a supercomponent.

- **Multi-domain modeling** — Not tied to any particular domain of engineering. Models can have a multidisciplinary nature.

- **Reusability** —Thanks to paradigms like encapsulation and causality, development of libraries with the intent of re-utilizing models is possible.

As this approach allows the user to build big and complex simulation models, one big advantage of *Amesim* is its solver because, generally, in a common software, the numerical method has to be chosen and the simulation parameters have to be tuned. On the contrary, this software contains 30 different numerical algorithms and automatically selects the one best adapted to the set of equations it has to solve [19].

Among many libraries that compose *Amesim* there is one, specially suited for engine simulation, developed at IFP Energies nouvelles (French Institute of Petroleum), called IFP-Engine, mainly composed by 0D/1D submodels. It provides the capability to generate various types of internal combustion engine models: all types of cylinder positioning, port-fuel or direct injection, CI (compression ignited) engines or SI (spark-plug ignited) engines, two and four stroke engines, among other characteristics.

2.5.3 Braking system modeling

As for engine modeling, hydraulic braking systems can also have different degrees of complexity depending on the type of analysis required. To reach an efficient simulation, the model has to be detailed enough to represent the dynamics of interest, but with most simple possible layout. Usually motorcycle braking systems are composed by several components, namely the master cylinders, calipers, brake pads and discs brakes, with the possibility of having ESP (electronic stability program) and ABS (anti-lock braking system) hydraulic modulator valves, all connected through a piping system [36]. These can be modeled with mathematical representations of the hydraulic characteristic functions of pumps, accumulators, valves and hydraulic cylinders or taking also into account the geometry and physical properties of each component (inertia, fluid compressibility, mass, spring, stiffness, etc). Hydraulic pipes/lines, which connect the different braking system components, can also be modeled with different degrees of complexity or not modeled at all (direct connection between components). In reality, these pipes are filled with fluid which are not infinitely stiff. Therefore it makes sense to consider compressibility of the liquid and friction (pressure losses) within the pipes. It may also be taken into account inertia effects if the study of transient behavior of the system and wave propagation along the network is necessary. When accounting with inertia, 3 types of models can be adopted [37]:

- **Lumped models (0D models)** — pressure and flow rate variation along the pipes are not significantly.
- **Distributive models (1D models)** — pressure and flow rate is computed at different locations along the pipes. It is a series of 0D models connected together.
- **CFD-1D models** — Using 1D Navier-Stokes equations, this type of models computes wave effects with a high level of accuracy.

As for the engine modeling, in this dissertation, the software used to create hydraulic braking systems was *Simcenter Amesim*. The use of this software gives the possibility to analyze and compare different hydraulic architectures, to evaluate braking distances or vehicle stability if connected to a vehicle model. It can also handle different types of braking systems either pneumatic or hydraulic for many different types of vehicles (trucks, cars, motorcycles, buses, trains, etc). Checking cavitation problems in the master cylinder is also possible depending on the selected complexity of the fluid properties [38]. There are three main libraries used for hydraulic modeling [39]:

- **HYD (Hydraulic library)** — Main library with indispensable components for system simulation. Some components are always needed when using the other two libraries. The components are defined by their hydraulic characteristics, usually provided by the component supplier.
- **HCD (Hydraulic component design library)** — A more flexible library made of basic elements and physical functions used to build detailed models that the previous library can't. The library's models are based on geometry and physical properties of the component trying to be replicated.
- **HR (Hydraulic resistance library)** — designed for the evaluation of pressure losses and flow rate distribution in hydraulic networks.

Chapter 3

Siemens eCVT test bench

Vehicle transmission researchers have always been developing new ways of reducing energy consumption as a consequence of the increasingly environmental concerns, pushing forward to new improvements in terms of reducing exhaust emissions and increasing vehicle efficiency. In the two wheeler's field these concerns also arise. One type of transmission that still didn't fully reached its potential, mainly used in scooters but also seen in bigger motorcycles, is the Continuously Variable Transmission (CVT). It has been an emerging transmission technology, not only for motorcycles but for other sorts of vehicles, offering continuum transmission ratios between desired limits, enhancing fuel economy and matching the engine operating conditions to the various driving scenarios [40]. In order to study a motorcycle's CVT behavior, a test bench was developed at Siemens' testing facilities centered on an Aprilia Mana 850 GT motorcycle's (Fig. 3.1) transmission, specifically called eCVT as it is electronically controlled (electronically controlled Continuously Variable Transmission).

Figure 3.1: Aprilia Mana 850 GT [41].

3.1 eCVT overview

The eCVT configuration consists in two pulleys with variable diameter connected by a rubber V-belt which allows motion transmission between them. Other configurations exist for different types of motorcycles where it is used a chain instead of a belt. Each pulley is composed by two parts: a fixed one, normally attached to the transmission shaft, and a movable one, which slides axially along the transmission shaft direction. Figure 3.2 identifies the principal eCVT components. The driving pulley movable part displacement determines the eCVT's transmission ratio as the belt position adjust to this change while never stopping to transmit movement. As stated previously, the driving pulley is controlled electrically by an electric motor actuation, which with the help of a double gear and a transmission shaft makes the control bushing spin. As the control bushing is attached to a thread, its rotation makes the driving pulley move in the axial direction. Figure 3.3 demonstrates the mentioned mechanism. On the other eCVT side, the driven pulley movable part position is not controlled directly by the electric motor. Instead, is controlled by a spring placed between the driven pulley and a centrifugal clutch (commonly used in this type of transmission), as shown in Fig. 3.4.

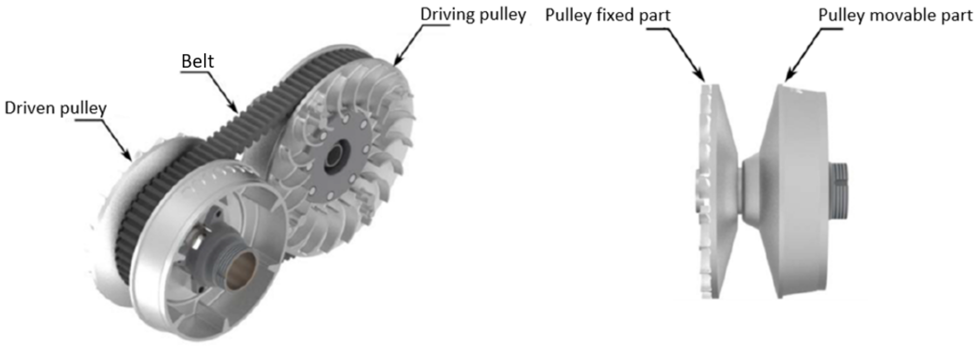


Figure 3.2: eCVT main components [42].

3.2 Test bench breakdown

For the test bench creation some adjustments to the eCVT had to be made and more components were also added to the system. Because the crankcase from the engine side wasn't needed, it was removed from the simulation. To simulate the engine crankshaft and the transmission output shaft behavior two specially made shafts were designed (Fig. 3.5). Each of the new specially made shafts is connected to an electric motor. The first electric motor is connected to the replaced engine crankshaft and is meant to simulate the combustion engine behavior, while the second one, connected to the transmission output shaft, is meant to simulate the motorcycle dynamics having into account the environment in which the motorcycle is passing by.

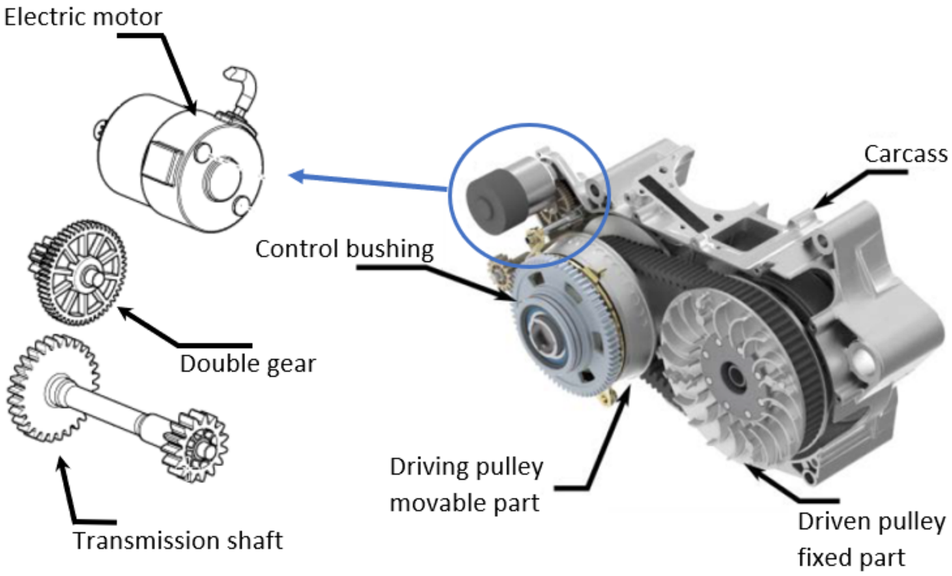


Figure 3.3: Transmission control mechanism [42, 43].

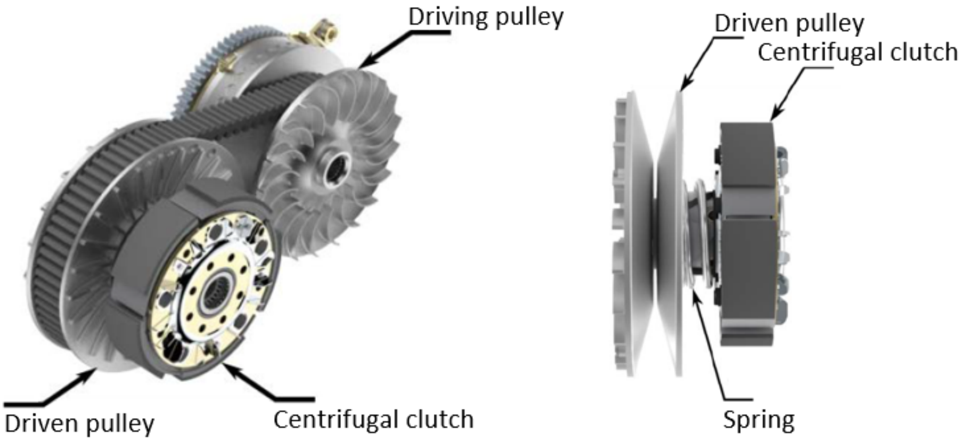


Figure 3.4: Driven pulley spring control [42].

Flange couplers were used to couple the shafts to their respective electric motor and 2 torque/speed KISTLER sensors were also placed giving the possibility to obtain the shafts' current torque or speed. Figure 3.6 gives an idea how the test bench is built based on the previous statements.

3.3 Main Goal: eCVT-in-the-loop

The test bench purpose is to reach the System-in-the-loop stage, or more specifically the eCVT-in-the-loop, applying the testing with simulation category from the model based system testing approach. In simpler words, the main focus is, with a real-time platform created and connected to the eCVT physical system, to create different motorcycle subsystems' models integrated in the

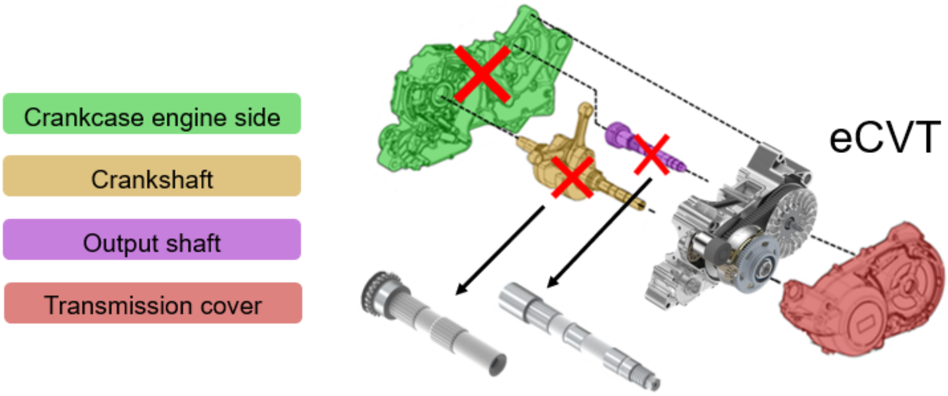


Figure 3.5: Crankcase removal and shafts replacement.

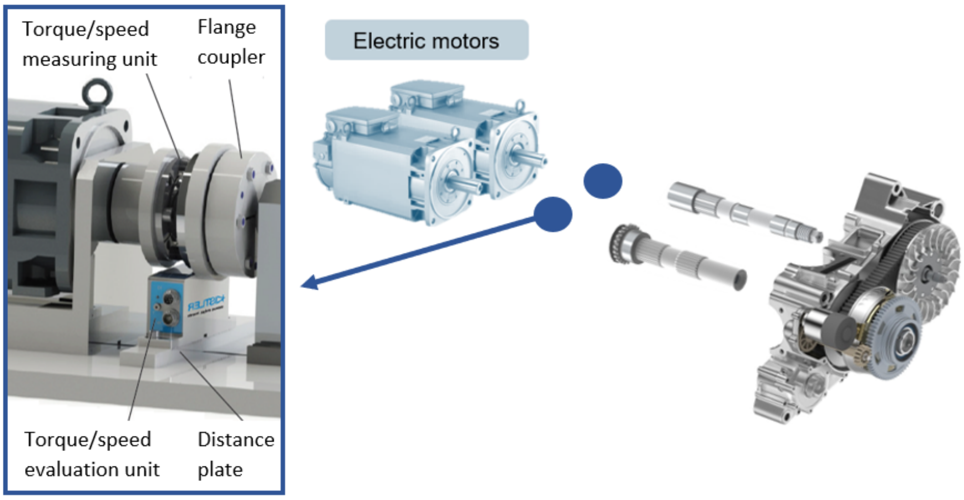


Figure 3.6: Flange coupler and torque/speed sensor placed in each of eCVT sides. Blue dot corresponds to one eCVT side [44].

testing loop. The final product will be a fully running real-time co-simulation around the Aprilia Mana 850 GT transmission.

Three different main simulations were made (Fig. 3.7 demonstrates how they relate with each other and the eCVT):

- **1D motorcycle engine model** — Resorting *Simcenter Amesim* software an Aprilia Mana 850 GT's engine model was created. The simulation receives a throttle command input by the user and calculates the torque generated by the engine crankshaft. The torque signal controls the first electric motor present in the eCVT, while the speed sensor, shown in Fig. 3.6, gives a feedback speed to the engine model.

- **Motorcycle multibody model** — Resorting *Simcenter 3D Motion* software an Aprilia Mana 850 GT's multibody model was created. The simulation executes a speed control on the second electric motor of the eCVT while receiving a torque feedback signal sent by a torque sensor placed in the output side of the eCVT.

- **1D motorcycle braking system model** — Resorting *Simcenter Amesim* software an Aprilia Mana 850 GT's braking system model was created. The simulation receives front and rear brake command inputs and generates a braking torque in the rear and front wheel of the motorcycle, which are sent to the motorcycle's multibody model. In return the multibody model sends an angular velocity feedback of each wheel to the braking system model.

When the real-time co-simulation is fully operational, the output coordinates of the motorcycle, for instance, the X,Y and Z coordinates and the yaw, pitch and roll angles, will be sent in real-time by the RT platform to another software called *Simcenter Prescan*, strictly used for visualization purposes. Important to state that this topic won't be subject of this dissertation.

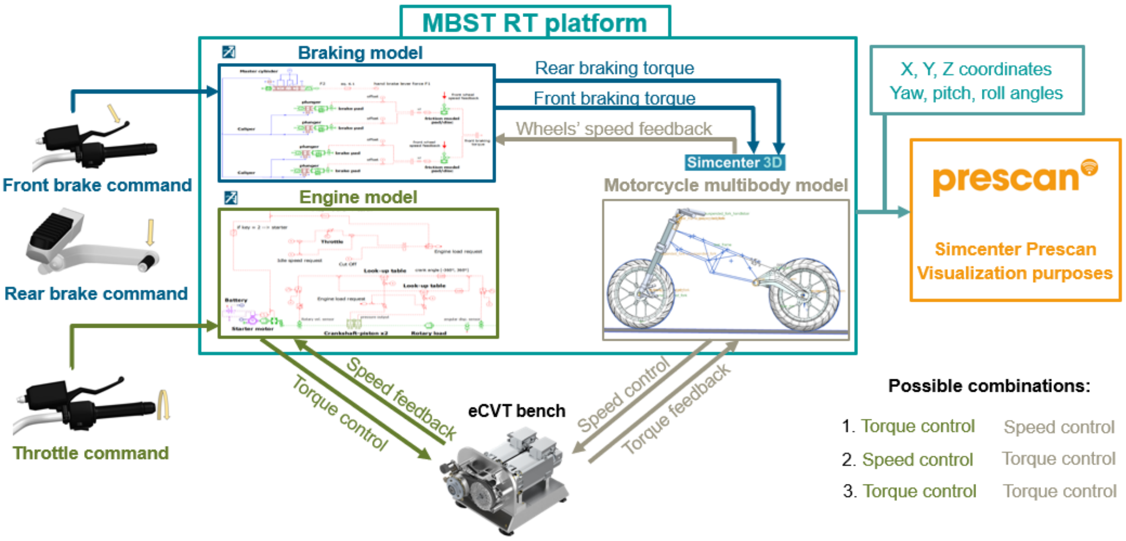


Figure 3.7: eCVT-in-the-loop co-simulation goal.

There are three different possibilities to run and control the eCVT test bench. The most suited and selected one was already explained above. It is also possible to control it with other combinations: Speed and torque control, or torque and torque control for the first and second electric motor respectively.

3.4 Real-time platform

The real-time platform elaboration is crucial to make real-time requirements possible because it works as the co-simulation orchestrator. Without it, there is no possibility to connect and integrate every simulation unit at stake. Its configuration integrates several elements including the eCVT test bench, as shown in Fig. 3.8:

- **PC** — Where the motorcycle models are developed and made suitable for real-time and offline analysis are performed. Test bench security and control configurations are also taken into account.
- **IPC** — The industrial PC is in charge of running the previous stated motorcycle models in real-time and perform all calculations to keep the co-simulation fully operational. The IPC employed was the *Siemens Simatic Industrial PC 627D*.
- **Data acquisition unit** — Responsible for data synchronization coming from sensors placed in the eCVT test bench and acquiring data for post processing analyses. The data acquisition unit employed was the *Simcenter Scadas Mobile*.
- **Control unit** — Controls directly the eCVT test bench electric motors.
- **EtherCAT slave** — Reads the data received and sends it towards the data acquisition and control unit. The selected type was the *EtherCAT Beckhoff EK1101*.

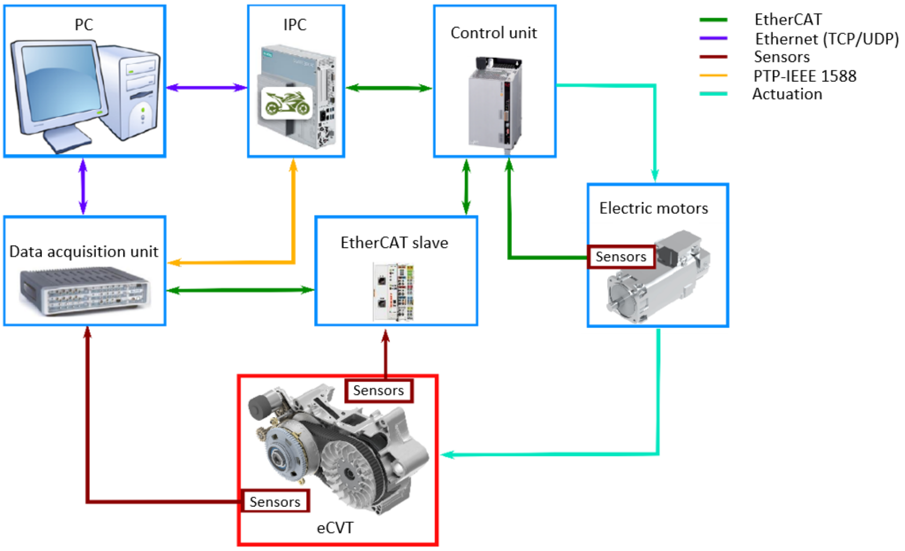


Figure 3.8: Real-time platform scheme [42].

Linux is used as the real-time operating system (RTOS), handling all real-time requirements in an adaptive version suited for the industrial PC [42].

Chapter 4

Motorcycle engine modeling

As previously mentioned in chapter 3, the work of this dissertation is centered around Aprilia Mana 850 GT motorcycle. The engine was the first part being modeled resorting the modeling tool *Simcenter Amesim* already introduced in section 2.5.2. A list of this program libraries were used for this construction process, namely the mechanical, the IFP (see section 2.5.2), signal and control, powertrain and electric motor and drives libraries.

It is important to mention that the created engine model is characterized by being a behavioral model as there were no technical or specific information and neither detailed 3D CAD engine models delivered by Aprilia brand along the process of its construction. With that being said, there was no access to the motorcycle detailed parameters and no possible model validation and comparison with accurate results. As a consequence, the only way to validate this model in the end is by verifying the results and check if it behaves like a normal motorcycle engine of this type. No problems should arise from this downside as the ultimate goal is not 100% accuracy but making a real-time co-simulation between motorcycle subsystems' models. The only available information about the motorcycle's engine is summed up at Tab. 4.1.

As previously mentioned in chapter 2, one of the simplest ways of constructing an engine model is making use of look-up tables resorting accurate experimental data. These tables' outputs can be connected to other components with higher complexity and predictability. In this case it was decided that the use of look-up tables would be needed, specifically to give the output pressures inside each motorcycle cylinder, where the inputs would be the current rpm's and angular displacement of the crankshaft. This turned out to be the best solution for the model creation by reducing as much as possible the model complexity even though there was no experimental data available to fill up the tables. The main reason for picking this kind of structure was to allow the model to have as output the engine torque depending on a throttle command, output to be sent to the eCVT bench as specified in Fig. 3.7 and to avoid complex control algorithms and data. Of this solution arose the necessity to find a way to create suitable data for the mentioned tables. To

Table 4.1: Engine characteristics [45].

Specification	**Description/Quantity**
Model	M290 Duel (Spark ignition engine)
Type	90° V-Twin, 4-stroke, 4 valves per cylinder
Camshaft	Single overhead camshaft, chain driven
Number of cylinders	2
Overall cylinder capacity	839.9 cm^3
Bore/stroke	88 x 69 mm
Compression ratio	10 - 10.6 : 1
Idle speed	1400 ± 100 rpm
Ignition	Starter motor
Injection system	Multipoint injection (port fuel)
Fuel	Premium unleaded gasoline

achieve this, before the creation of the main and final engine model, it was needed to create an auxiliary model that would compute data needed to fill the look-up tables.

4.1 Auxiliary engine model

For this model a set of 0D/1D submodels were connected together using mainly the IFP library trying to replicate all engine important parts. Differently from the final engine model, this one is characterized to be a steady state model where the only input needed, set by the user, is a fixed rpm at which the engine is running. All calculations in every submodel are performed based on the constant rpm value set.

4.1.1 Cylinder

4.1.1.1 Combustion model

Two possible types of models could be selected to simulate the combustion process inside each engine cylinder, either a non-predictive model or a predictive one. In a simplification attempt, although there is no experimental data, it was decided to go along with a non-predictive model called Wiebe, introduced by Ivan Wiebe. One reason for picking this approach was the fact that this type of model is particularly interesting in situations where the pressure curve inside the cylinder is not available due to difficulties of measurement or when the engine does not exist because is in the development phase. This model is characterized to be a continuous function which allows the computation of the burned mass fraction of fuel profile in relation to the crankshaft angle of internal combustion engines [32]. The continuous function, presented in equation 4.1, relates different variables, which result in a specific and characteristic S-shape profile, as demonstrated in Fig. 4.1 in each one of the traces presented in the graph. The rate at which fuel-air mixture burns increases from a low value intermediately following the spark discharge to a maximum about halfway through the burning process and then decreases to a close to zero value as the combustion process ends [46].

$$y_{burn} = 1 - exp\left(-a \cdot \left(\frac{\theta_c - \theta_0}{\Delta\theta}\right)^{m+1}\right) \tag{4.1}$$

Where,

θ_c	= crankshaft angular displacement
θ_0	= crankshaft angle corresponding to start of combustion
$\Delta\theta$	= total combustion duration
a	= combustion efficiency (fitting parameter)
m	= shape of the mass fraction burned profile (fitting parameter)

Figure 4.1 gives an example on how these above mentioned parameters can affect the Wiebe function. For instance, for constant combustion duration $\Delta\theta$ and a parameter, changing the m parameter changes the shape of the mass fraction burned profile. It is important to state that Wiebe function's parameters are normally tuned to best match experimental cylinder pressure. However, each set of parameters defined is only suitable for a single engine operation condition, thus at first hand when running under transient behavior deviation in the simulation can be introduced [34]. To surpass this disadvantage *Simcenter Amesim* has a tool which can tune Wiebe's parameters, defining a map covering the necessary engine operating conditions that best fits the different cylinder pressure profiles [33]. As there is no experimental data on cylinder pressure profiles tuning Wiebe's parameters is not possible either for single or multiple operation conditions (using

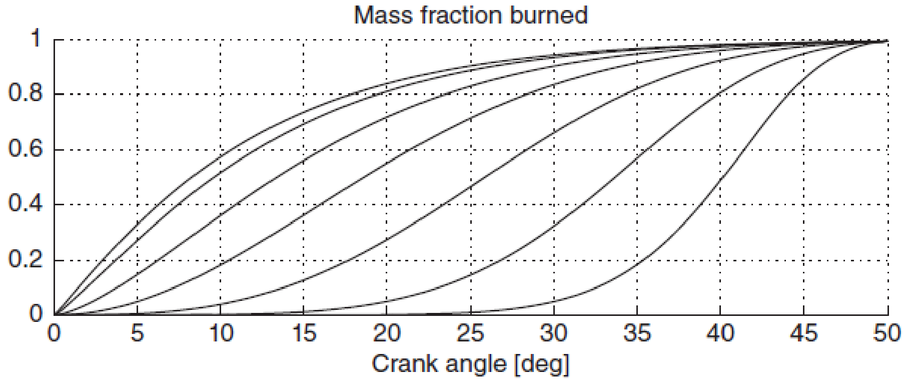


Figure 4.1: Burned mass fraction of fuel profile for different values o m parameter ($\Delta\theta$ = 50 degrees and a = 5) [47].

maps). Because of this, it was decided to use constant Wiebe's parameters for spark ignition engine according to [48] and use values from other engine models presented in [19], even though it will bring deviations when running different simulations. According to the previous information, the parameters have been fitted with a = 5, m = 2, $\Delta\theta$ = 35° and θ_0 = 338°. With the selected set of parameters the evolution of the burned mass fraction of fuel (y_{burn}) can be defined by equation 4.2:

$$\frac{y_{burn}}{d\theta_c} = a \cdot \frac{m+1}{\Delta\theta} \cdot \left(\frac{\theta_c - \theta_0}{\Delta\theta}\right)^m \cdot exp\left(-a \cdot \left(\frac{\theta_c - \theta_0}{\Delta\theta}\right)^m\right) \tag{4.2}$$

Knowing the burned mass fraction of fuel, the instantaneous heat released by the combustion could be characterized by:

$$\frac{dQ}{d\theta_c} = \frac{y_{burn}}{d\theta_c} \cdot Q_{total} \tag{4.3}$$

Where,
Q_{total} = total heat released by combustion

To get the total heat released by combustion, the equation 4.4 is used:

$$Q_{total} = m_{BG}^{fuel} \cdot LHV \tag{4.4}$$

Where,
m_{BG}^{fuel} = mass of fuel which will be burned:
is equal to $mass_{fuel}$ (gas inside the cylinder) if mixture is lean
or equal to $mass_{air}$ (gas inside the cylinder) / AFR_{stoich} if mixture is rich
LHV = lower heating value of fuel

Stoichiometric air/fuel ratio (AFR_{stoich}) and lower heating value (LHV) of fuel depend respectively on the chemical composition and gas properties of the fuel. In order to obtain values to this properties the engine fuel was defined as a standard unleaded gasoline (submodel provided by *Simcenter Amesim*) with a generic chemical composition in the form of $C_xH_yO_z$ where x = 6.76, y = 12.48 and z = 0.08. Value for the lower heating value (LHV) used was equal to 42 MJ per kg of fuel according to [49]. With the fuel properties set is possible to obtain the stoichiometric air/fuel ratio (AFR_{stoich}) according to equation 4.5, equal to 14.2341:

$$\left(\frac{A}{F}\right)_{stoich} = \frac{\left(x + \frac{y}{4} - \frac{z}{2}\right) \cdot (M_{O_2} + x_{N_2O_2} \cdot M_{N_2})}{x \cdot M_C + y \cdot M_H + z \cdot M_O} \tag{4.5}$$

Where,
$x_{N_2O_2}$ = N_2/O_2 molar fraction in air
M_i = molar mass of different species, as O_2, N_2, C, H and O

At each injection cycle the cylinder receives the crankshaft angular displacement (θ_c) relative to time (calculation process shown in section 4.1.1.3), the intake and exhaust gas mass fractions of the mixture (composed by burned gases, air and vaporized fuel) and the intake and exhaust mass flow rate of the mixture, which finally allow the program to compute the mass evolution of the 3 components inside the cylinder, using equations 4.6, 4.7, 4.8:

$$\left(\frac{dm_{fuel}}{dt}\right)_{combustion} = -\frac{1}{LHV} \cdot \left(\frac{dQ_{comb}}{dt}\right) \tag{4.6}$$

Using stoichiometric air/fuel ratio (AFR_{stoich}), the evolution of the mass of fresh air is defined.

$$\left(\frac{dm_{air}}{dt}\right)_{combustion} = -\frac{AFR_{stoich}}{LHV} \cdot \left(\frac{dQ_{comb}}{dt}\right) \tag{4.7}$$

The evolution of the mass of burned gases is defined by the mass conservation equation.

$$\left(\frac{dm_{BG}}{dt}\right)_{combustion} = -\left(\frac{dm_{fuel}}{dt}\right)_{combustion} - \left(\frac{dm_{air}}{dt}\right)_{combustion} \tag{4.8}$$

4.1.1.2 Pressure dynamics

For pressure behavior inside the motorcycle's cylinders it was applied a function that can be used for both fixed or variable volume chambers with or without heat exchange. For this case, the function is suited for a variable volume chamber (as the volume inside the cylinders change with different piston positions). Regarding cylinder walls heat exchange there were several possibilities to be chosen: Eichelberg model, Annand model or Whoschni model (the most suited for engine applications) [34]. However, for the sake of simplicity an adiabatic wall heat exchange model was selected.

The process to arrive to the above mentioned function is reached by differentiating the ideal gas relation $P = \rho RT$:

$$\frac{dP}{dt} = \rho \cdot \left(\frac{dR}{dt} \cdot T + R \cdot \frac{dT}{dt}\right) + r \cdot T \cdot \sum \frac{d\rho_i}{dt} \tag{4.9}$$

Where,

R = perfect gas constant of a N gas mixture
T = combustion chamber temperature
ρ = mixture density

To solve this equation the terms T and $\frac{dT}{dt}$, R and $\frac{dR}{dt}$, ρ and $\sum \frac{d\rho_i}{dt}$ must be defined. The temperature T inside the combustion chamber can be simply calculated resorting the ideal gas relation.

On the other side, temperature evolution $\frac{dT}{dt}$ can be defined by equation 4.13. The steps to reach this equation are the following:

Using the first law of thermodynamics the variation of the internal energy U applied to an open system can be described as in equation 4.10.

$$\frac{dU}{dt} = \sum \dot{m}_i \cdot h_i + \frac{dQ}{dt} - \frac{dW}{dt} \tag{4.10}$$

Where,

$\dot{m}_i \cdot h_i$ = enthalpy flow rate at a port

$\frac{dQ}{dt}$ = heat flow provided or evacuated from the control volume (for adiabatic model this parcel is always 0)

$\frac{dW}{dt}$ = work done by the system, which can also be expressed dependent of volume variation $\frac{dW}{dt} = -P \cdot \frac{dV}{dt}$

As the mixture is composed by several gases internal energy can be defined by equation 4.11, where x_i is the mass fraction of a gas in the mixture.

$$U = \sum m_i \cdot u_i = m \sum u_i \cdot x_i \tag{4.11}$$

Integrating equation 4.11 and merging it with equation 4.10, we got equation 4.12.

$$m \cdot C_v \cdot \frac{dT}{dt} + m \sum \frac{dx_i}{dt} \cdot u_i + \frac{dm}{dt} \int C_v \cdot dT = \sum \dot{m}_i \cdot h_i + \frac{dQ}{dt} - \frac{dW}{dt} \tag{4.12}$$

The evolution of combustion chamber temperature can now be calculated, manipulating equation 4.12 to the following form:

$$\frac{dT}{dt} = \frac{\sum \dot{m}_i \cdot h_i + \frac{dQ}{dt} - \frac{dW}{dt} - m \sum \frac{dx_i}{dt} \cdot u_i - \frac{dm}{dt} \int C_v \cdot dT}{m \cdot C_v} \tag{4.13}$$

The perfect gas constant of a mixture, R, that appear in equation 4.9, can be calculated in the following manner:

$$R_{mixt} = \sum_{i=1}^{N_{gas}} x_i \cdot r_i \tag{4.14}$$

Where,

x_i = mass fraction of gas i

r_i = perfect gas constant R of gas i

Being known that x_i presented in equation 4.14 can be shown in the form of $\frac{\rho_i}{\rho}$, perfect gas constant temporal evolution, $\frac{dR}{dt}$, in its turn, can be calculated using equation 4.15.

$$\frac{dR_{mixt}}{dt} = \sum_{i=1}^{N_{gas}} \frac{dx_i}{dt} \cdot r_i = \sum_{i=1}^{N_{gas}} \left(\frac{\dot{\rho}_i}{\rho} - \rho_i \cdot \frac{\dot{\rho}}{\rho^2} \right) \cdot r_i \tag{4.15}$$

The density evolution of gas i in the mixture $\dot{\rho}_i$, presented in equation 4.15 can be defined according to the mass flow rate at the intake, exhaust and in the combustion process, as presented in equation 4.16, and with the evolution of the combustion chamber volume defined in section 4.1.1.3.

$$\frac{dm_i}{dt} = \left(\frac{dm_i}{dt} \right)_{ports} + \left(\frac{dm_i}{dt} \right)_{combustion} \tag{4.16}$$

4.1.1.3 Architecture and geometry

Some important variables sent to other submodels are deduced from the cylinder and crankshaft geometry and engine architecture. Figure 4.2 shows the main defined characteristics regarding this topic. According to [50], the crankshaft has paired crank throws. However, with no physical parts available there is no possibility to know if there is a piston pin offset (see Fig. 4.2, $dpiston$) or crankshaft offset (see Fig. 4.2, $dcrank$), so they were both set to 0. All other variables can be taken from Tab. 4.1 except from crank radius which is equal to $stroke/2$ when offsets are set to 0 and connecting rod length which was set to 104 mm as it was unknown (this value is normally 1.4 to 2.2 the stroke length).

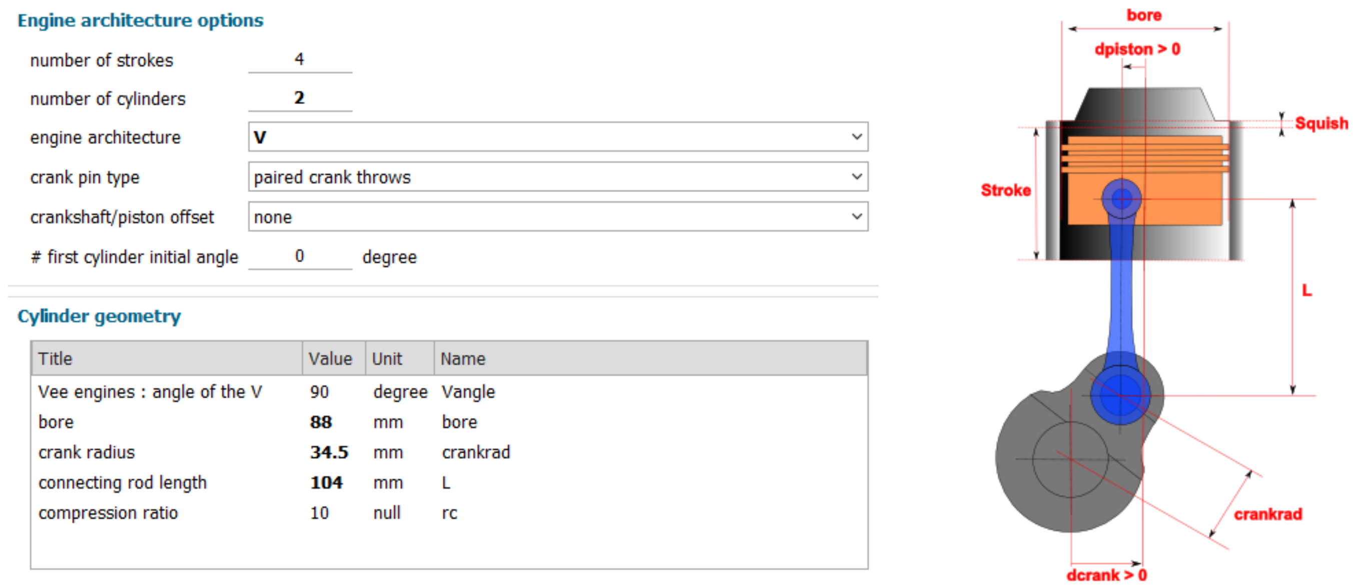
The first calculated variable is the crankshaft angular displacement θ, by simply integrating the rotary velocity of the crankshaft which in this specific auxiliary model is set constant throughout the simulation. The piston position s relative to the piston frame shown in Fig. 4.3, can be formulated as [51]:

$$L^2 = R^2 + s^2 - 2R \cdot s \cdot cos(\theta) \tag{4.17}$$

Where,

L = length of the connecting-rod

R = crank radius



Engine architecture options

number of strokes 4

number of cylinders **2**

engine architecture **V**

crank pin type paired crank throws

crankshaft/piston offset none

first cylinder initial angle 0 degree

Cylinder geometry

Title	Value	Unit	Name
Vee engines : angle of the V	90	degree	Vangle
bore	**88**	mm	bore
crank radius	**34.5**	mm	crankrad
connecting rod length	**104**	mm	L
compression ratio	10	null	rc

Figure 4.2: Engine geometry and architecture

Manipulating equation 4.17 we get:

$$s = R \cdot cos(\theta) + \sqrt{L^2 - R^2 \cdot sin^2(\theta)} \tag{4.18}$$

Defining the piston position, it is possible to formulate the volume inside the combustion chamber as a function of the crankshaft angular displacement:

$$V = \frac{\pi \cdot B^2}{4} \cdot (L + R - s(\theta)) \tag{4.19}$$

The total volume of the chamber is calculated by summing the volume derived from the piston position in equation 4.19 and a dead volume (clearance volume). The clearance volume V_c is equal to the volume of the combustion chamber when the piston is at the top dead center and can be calculated as presented in equation 4.20:

$$V_c = \frac{V_d}{rc - 1} \tag{4.20}$$

Where,

V_d = overall cylinder capacity or maximum volume displaced by the piston

rc = compression ratio

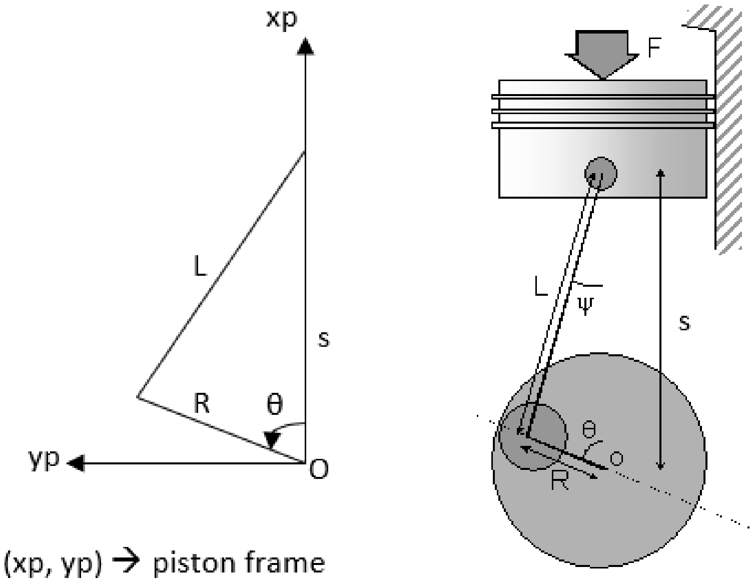


Figure 4.3: Crankshaft and connecting-rod dimensions

Piston linear velocity is characterized by equation 4.21:

$$V = G \cdot w \tag{4.21}$$

Where,

G = lever arm done by the connecting-rod associated to the crank throw

w = rotary velocity of the crankshaft

The variable G can also be seen as the piston velocity with respect to the crankshaft angular displacement. So differentiating equation 4.18, we get [51]:

$$|G| = R \cdot sin(\theta) + \frac{R^2 \cdot sin(\theta)cos(\theta)}{\sqrt{L^2 - R^2 \cdot sin^2(\theta)}} \tag{4.22}$$

4.1.1.4 Force and torque generated

The force F applied on the piston (see Fig. 4.3) can be calculated as follows:

$$F = (P_{cyl} - P_{atm}) \cdot S_p \tag{4.23}$$

Where,
P_{cyl} = pressure inside the combustion chamber computed in section 4.1.1.2
P_{atm} = atmospheric pressure
S_p = piston surface area

Furthermore, the torque generated by each cylinder submodel in the crankshaft is obtained from equation 4.24, using the lever arm G already explained in section 4.1.1.3.

$$T = G \cdot F \tag{4.24}$$

4.1.2 Valvetrain and cylinder head

4.1.2.1 Valve's flow characterization

The submodel defined for this purpose uses look-up tables to determine the intake and exhaust valve's lift as function of the crankshaft angle. As once again there is no detailed parameters of the engine valvetrain, data from a previously made gasoline engine model was used [19]. This look-up tables or curves not only define the valve lifting as function of the crankshaft displacement but also the valve timing between the 2 exhaust and 2 intake valves for each cylinder, where variables like IVO (intake valve opening), IVC (intake valve closing), EVO (exhaust valve opening), EVC (exhaust valve closing) and overlap are defined. The curves are shown in Fig. 4.4, where BDC and TDC represent bottom dead center and top dead center respectivelly. The flow through intake and exhaust valves is also characterized with a function which evaluates the mass flow and enthalpy flow rate through an orifice. The mass flow rate is computed as follows:

$$\dot{m} = C_q \cdot A \cdot C_m \cdot \frac{P_{up}}{\sqrt{T_{up}}} \tag{4.25}$$

Where,
C_q = discharge coefficient
A = orifice area
C_m = mass flow parameter
P_{up} = upstream pressure
T_{up} = upstream temperature

The discharge coefficient, used to introduce extra losses due to local friction and loss of kinetic energy, and the orifice area can be expressed as the effective area as $A_e = C_d \cdot A$. Putting it into words it can be said that the effective area is the outlet area of an ideal friction-less nozzle in which would pass the same flow, with the same pressure drop, with uniform constant-pressure flows upstream and downstream [49]. Once again, experimental results from the same gasoline engine model were taken to define curves for the intake and exhaust inflow effective area as function of

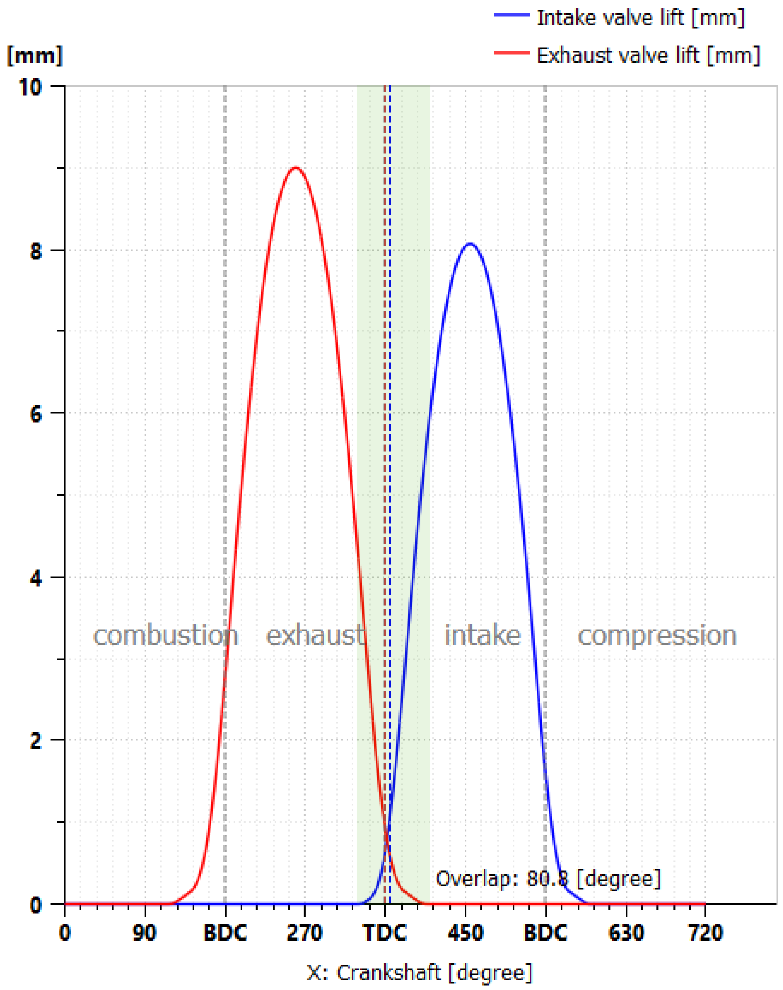


Figure 4.4: Intake and exhaust valve lifting curves.

the valve lift. The curves are shown in Fig. 4.5. The final value of the intake inflow effective area and exhaust outflow effective area are reached after multiplying the value taken from the curves with the number of exhaust and intake valves for each cylinder. The flow parameter C_m, which depends on the pressure ratio P_{down}/P_{up}, where P_{down} is the downstream pressure, can be calculated in two possible ways either if the flow is sonic or subsonic. The flow is sonic if the pressure ratio is below or equal to the critical pressure and is subsonic if the pressure ratio is bigger than the critical pressure. Critical pressure is shown in equation 4.26, where γ is the specific heat ratio.

$$P_{cr} = \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}} \tag{4.26}$$

So the mass flow rate can be computed as equation 4.27.

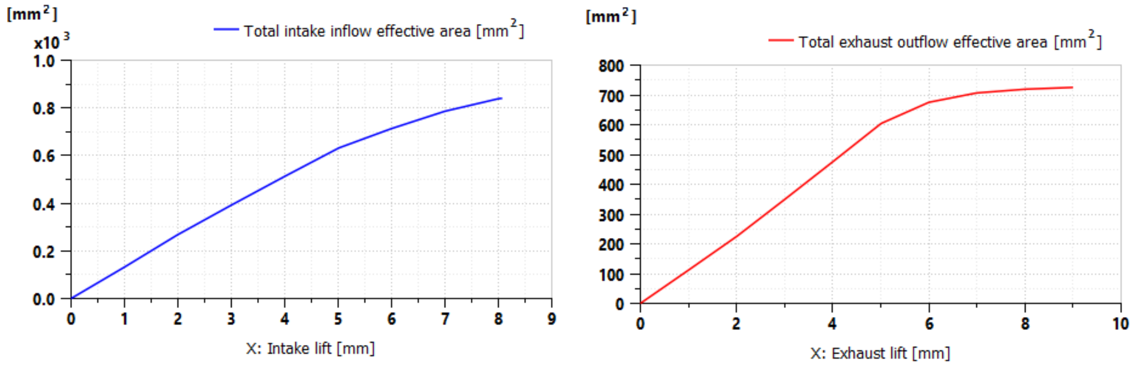


Figure 4.5: Intake and exhaust effective area as function of valve lift.

$$C_m = \begin{cases} \sqrt{\frac{2\cdot\gamma}{r\cdot(\gamma-1)}} \cdot \sqrt{\left(\frac{P_{dn}}{P_{up}}\right)^{\frac{2}{\gamma}} - \left(\frac{P_{dn}}{P_{up}}\right)^{\frac{\gamma+1}{\gamma}}}, & \text{if } \left(\frac{P_{dn}}{P_{up}}\right) > P_{cr}(subsonic) \\ \\ \sqrt{\frac{\gamma}{r} \cdot \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}}, & \text{if } \left(\frac{P_{dn}}{P_{up}}\right) \leq P_{cr}(sonic) \end{cases} \tag{4.27}$$

Upstream and downstream pressure can be considered as gas pressure inside the cylinder or gas pressure coming to the cylinder depending on the exhaust or intake mass flow rate calculation.

Now that the mass flow rate is computed, the enthalpy flow rate is expressed as equation 4.28.

$$dh = \dot{m} \cdot h(P_{up}, T_{up}) \tag{4.28}$$

4.1.2.2 Firing order

To set the opening of intake and exhaust valves of one cylinder related to the other, the engine firing order must be defined. For a 90º V-twin engine a simplistic information about this topic would be that the firing order is 1-2-1. However, this notation is inadequate because there are 2 different firing scenarios with this kind of order. The first one, used for example by the Ducati 955 cm^3 V-twin racing engine, can be formulated as follows:

Imagining that cylinder 1 is at top dead center (TDC) during combustion. Because cylinders are disposed at a mutual 90º and the connecting rods are on a common crankpin, cylinder number 2 is at 90º before top dead center (BTDC) on one of its strokes. In this case it is at 90º before top dead center (BTDC) on the exhaust stroke, so cylinder number 2 will fire when cylinder number 1 is at 450º after top dead center (ATDC) on its 720º cycle.

In the other case cylinder number 2 is at 90º before top dead center (BTDC) on the compression stroke which would mean that the firing position for cylinder number 2 is when cylinder number 1 is at 90º after top dead center (ATDC).

The first case was selected as the second would make the motorcyclist experience big vibrations on the rear end because there is only a 90º crankshaft rotation between firing of cylinder number 1 and 2 [52]. Figure 4.6 represents the firing order chosen.

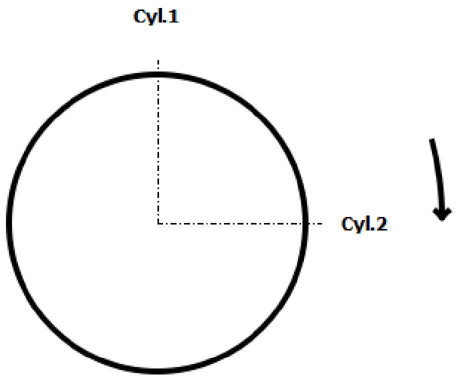


Figure 4.6: Engine firing order diagram. Cylinder 1 is the reference: cylinder 1 fires at 0º and cylinder 2 at 450º

4.1.3 Engine Friction

A useful parameter to measure engine's ability to do work is the mean effective pressure (MEP) which is defined as:

$$MEP = \frac{\text{work produced per cycle}}{\text{volume displaced per cycle}} \tag{4.29}$$

The friction mean effective torque (FMEP) can be calculated inserting the friction torque into the definition of MEP [47], as represented in equation 4.30.

$$FMEP = \frac{T_{frict} \cdot n_{strokes} \cdot \pi}{V_{cyl} \cdot n_{cyl}} \tag{4.30}$$

Where,

T_{frict} = friction torque

$n_{strokes}$ = number of strokes

V_{cyl} = volume displaced per cylinder

n_{cyl} = number of cylinders

In this submodel there is a possibility to define FMEP resorting data maps or by expressions. For this dissertation it was adopted an expression where FMEP is calculated as function of engine crankshaft rpm, presented in equation 4.31, taken from [46], where several tests were performed on four-stroke cycle four cylinder SI engines between 845 and 2000 cm^3 displacement at wide-open throttle (WOT).

$$FMEP = 0.05 \cdot \left(\frac{rpm}{1000}\right)^2 + 0.15 \cdot \left(\frac{rpm}{1000}\right) + 0.97 \tag{4.31}$$

With FMEP calculation in equation 4.31 depending on the constant rpm value set, the friction torque calculation will be possible resorting equation 4.30.

4.1.4 Intake and exhaust manifold

For these submodels the function presented in section 4.1.1.2 can be also applied. However instead of a variable volume chamber these ones are characterized to have fixed volume. It is also possible to impose a heat flow rate in the chamber due to convective exchange between fluid/walls. In every port of the chamber the enthalpy, the mass flow rate of the mixture and the mass fractions of the mixture's components is received as an input and as consequence the temperature, pressure and density are calculated, also according to section 4.1.1.2. Based on 3D CAD data the intake manifold has an approximate total volume of 0.716 L and the exhaust one a total of 0.248 L. Figure 4.7 shows the motorcycle intake manifold.

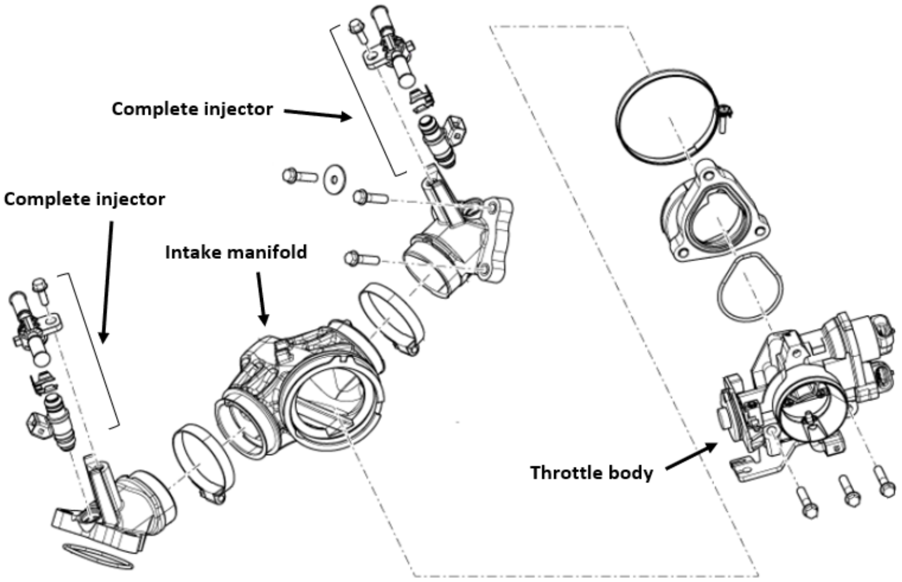


Figure 4.7: Intake manifold, injectors and throttle body representation [43].

4.1.5 Throttle body

The flow through the throttle body can be characterized using the same equations for mass and enthalpy flow rate as in section 4.1.2.1, based on Barré de Saint Venan equation which considers the throttle body as an isentropic nozzle [29]. However, for this submodel instead of defining the throttle valve effective area trough look-up tables it is used the throat area (A), computed resorting geometric dimensions, and the discharge coefficient (C_q). For the discharge coefficient (C_q), to simplify the process it was assumed a constant value of 1 (no losses), despite being possible to use experimental maps which give the discharge coefficient as function of the throttle angle and pressure ratio [29, 52]. The throat area (A) is the minimal geometric area of flow across the throttle valve. It is calculated by projecting the elliptic surface of the throttle valve onto the vertical plane [29] and is expressed as equation 4.32.

$$A = \frac{\pi \cdot D^2}{4} \cdot \left[1 - \frac{cos(\theta)}{cos(\theta_0)} + \frac{2}{\pi} \cdot \left(\frac{a}{cos(\theta)} \cdot \sqrt{cos^2(\theta) - a^2 \cdot cos^2(\theta_0)} + \right.\right.$$
$$\left.\left. + \frac{cos(\theta)}{cos(\theta_0)} \cdot arcsin\left(a \cdot \frac{cos(\theta_0)}{cos(\theta)} \right) - a \cdot \sqrt{1 - a^2} - arcsin(a) \right) \right] \tag{4.32}$$

Where,

θ = current throttle angle

θ_0 = throttle angle when A = 0

a = d/D ratio

d = shaft diameter of the throttle

D = bore diameter of the throttle

The throttle body can be seen in Fig. 4.7 and the previous equation's variables in Fig. 4.8. The equation 4.32 passes to the imaginary domain when $(cos^2(\theta) - a^2 \cdot cos^2(\theta_0)) < 0$, giving a threshold angle at which the area A is saturated. The threshold angle is $\theta_{max} = arccos(a \cdot \theta_0)$ and the saturated throat area is equal to:

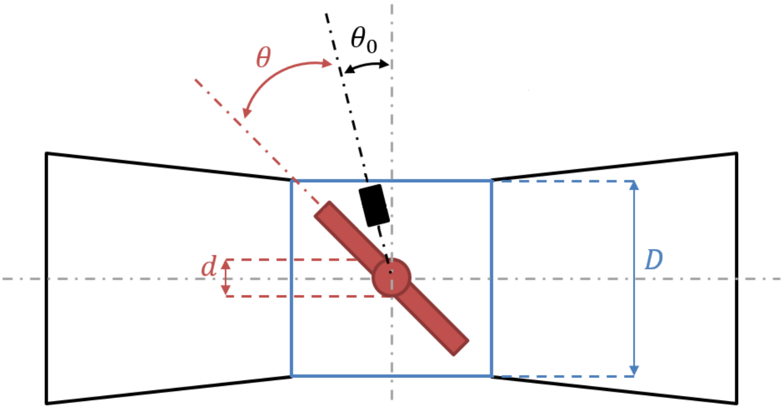


Figure 4.8: Throttle variables.

$$A = \frac{\pi \cdot D^2}{4} \cdot \left(1 - \frac{2 \cdot a}{\pi} \cdot \sqrt{1 - a^2} - \frac{2 \cdot arcsin(a)}{\pi} \right) \tag{4.33}$$

Since the throttle angle θ_0 was set to 3° and all simulations of this auxiliary engine model were performed with wide open throttle (WOT), $\theta = 90°$, the threshold angle is reached and the throat area can be calculated as in equation 4.33. Bore diameter (D) was set to 38 mm according to [45] and shaft diameter of the throttle was set to 3 mm. The final throat area computed is 1020.23 mm^2.

4.1.6 Injection

Table 4.1 and Fig. 4.7 show that the type of injection system used by the Aprilia Mana 850 GT is the port fuel injection, where a fuel injector is located at every intake port near the intake manifold [53]. A single injecting pulse per cycle is adopted opposing the multiple pulses scenario which is often used in direct injection engines [31]. The submodel implemented has the purpose to generate a simple injection pulse with a trapezoidal mass flow rate profile, as presented in Fig. 4.9.

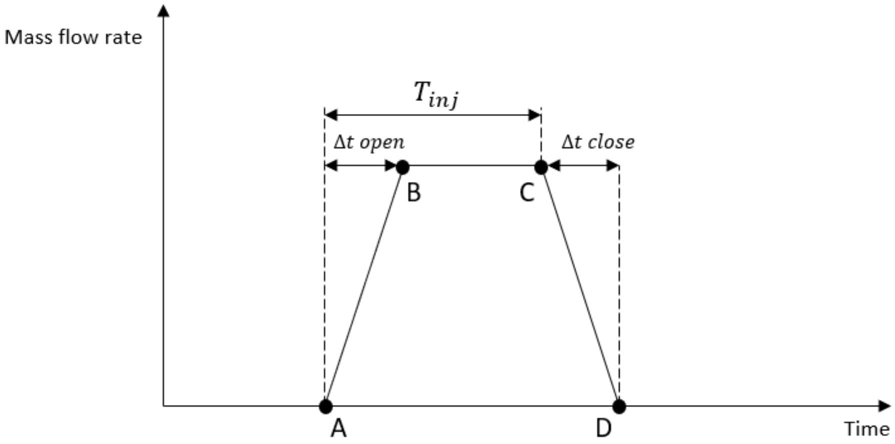


Figure 4.9: Injection pulse trapezoidal profile.

4 important points are defined in Fig. 4.9 related to injection parameters:

- **Point A** — is the point where injection starts and its related with injection timing. It is advantageous to inject fuel just before the inlet valve opening such that sufficient time for fuel evaporation is available [31].
- **Point B** — is defined by the maximum static mass flow rate set to 0.048 kg/s. Also characterizes the time opening phase of the injector needle (abcissa of B relative to A) set to a very small value of 0.1 µs.
- **Point C** — determines the injection duration (abcissa of C relative to A). This value will not be constant and will be changing for different rpm set in the simulation, as will be explained in section 4.1.7.2.
- **Point D** — determines the time closing phase of the injector needle (abcissa of D relative to C). Also set to 0.1 µs.

For this dissertation no evaporation model in the fuel ports was taken into account because gaseous fuel injection is used instead of liquid fuel injection. Enthalpy flow rate is calculated as function of mass flow rate and the specific fuel enthalpy which depends on fuel temperature and pressure.

4.1.7 Engine control

4.1.7.1 Normal approach

Today's SI engines have about 15-25 sensors and 6-8 main manipulated variables which are controlled with a powerful microcomputer control unit (ECU) with 80-120 look-up tables and many control algorithms [54]. There are even additional maps that handle effects like cold start, hot/cold climate or high altitude compensation [47]. Gasoline engine control can be subdivided in torque control, emission control, idle speed control, coolant temperature control and lubrication control. Focusing on what matters more for this auxiliary engine model, the torque control can be also segmented in the following control types [54]:

- **Fuel injection control** — ECU calculates the required fuel quantity as per engine speed and load at a particular point [53]. For example, Fig. 4.10 shows the injection pulse width per load % and rotational speed of a spark plug engine.

- **Air/fuel ratio control** — For an optimal combustion (engine performance, fuel consumption and emissions) is needed a close to stoichiometric mixture of air and fuel in the range $0.8 < \lambda < 1.4$ and because of the mostly applied three-way catalyst the air/fuel ratio must be precisely in the range of $0.98 < \lambda < 1.02$, where λ is the ratio between current AFR and the stoichiometric AFR [54]. When air/fuel ratio control and lambda sensor feedback is used we have a closed loop control, using air/fuel ratio tables and several correction factors and algorithms [55].

  One of the major elements for AFR control is mass air flow estimation through the intake manifold. This can be done by several techniques: direct air mass flow measurement using hot film/hot wire air mass flow meter or MAF (Mass air flow sensor), speed density method using intake manifold pressure using MAP (Manifold absolute pressure sensor) and alpha-N method based on throttle position using TPS (throttle position sensor). Second approach also needs volumetric efficiency look-up tables to calculate the mass air flow [55, 56].

- **Ignition control** — The open-loop scheme for the ignition timing control also relies on pre-determined look-up tables depending on the engine speed and load (as measured by manifold pressure or other related signals). Some correction factors are also applied, for example, based on the temperature of the engine coolant [56].

- **Knock control** — the optimum spark timing is often accompanied by the occurrence of knock in the combustion and even leads to engine damage in extreme conditions. Therefore, a trade-off is necessary in engine control between maximizing the spark advance and avoiding the occurrence of engine knock [56].
- **Throttle opening angle** — is controlled depending on the torque demand by the driver through signals from the accelerator pedal sensor and other system requirements [56].

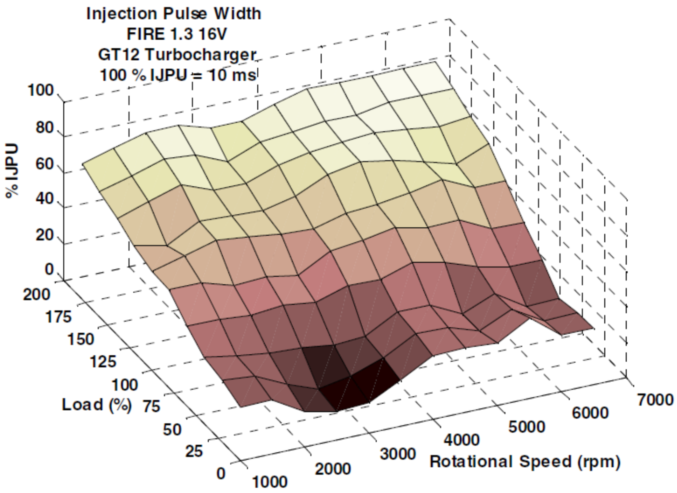


Figure 4.10: Injection pulse width per load % and rotational speed 2D look-up table [57].

4.1.7.2 Approach adopted

Previous section 4.1.7.1 shows how complex and detailed engine control systems can be and how much experimental and tuning data is needed. As there is no information available from Aprilia regarding this complex topic, throughout sections 4.1.1.1 to 4.1.6 it is shown that possible control variables are made constant instead of being computed with the help of look-up tables and control algorithms. The only credible data available regarding engine performance was found at [58] in the form of a graph where engine torque is characterized as function of engine rpm. Graph was adapted and presented in Fig. 4.11. Supported by this data a very rough approach was used: for an engine speed (rpm) set before running the auxiliary engine simulation (steady state model where engine rpm are constant throughout the simulation), the fuel injection duration specified in section 4.1.6 was tuned in order to give the respective engine torque presented in Fig. 4.11. This process was performed from 1000 to 8500 engine rpm obtaining a specific fuel injection duration needed for each of the values of engine speed set.

4.1.8 Model overview and look-up table construction

The auxiliary engine model is finally presented in Fig. 4.12. Having fully assembled this model,

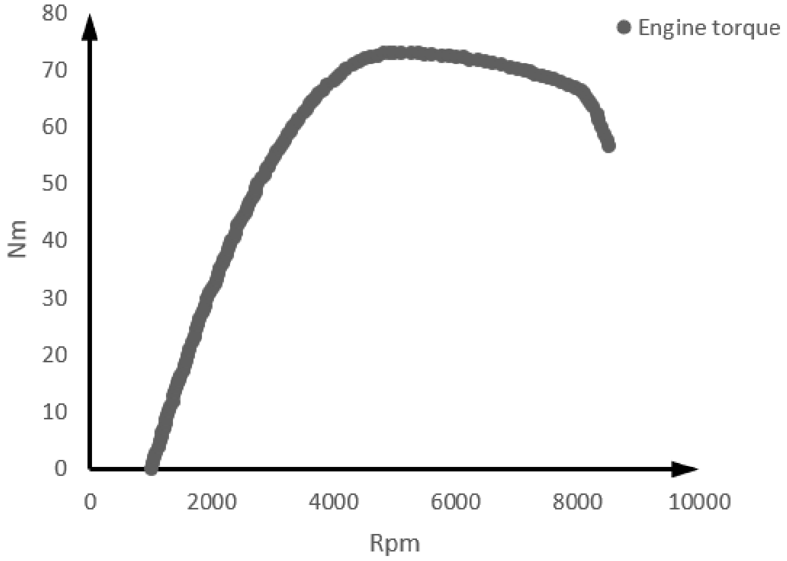


Figure 4.11: Aprilia Mana 850 GT's engine torque as function of crankshaft angular velocity [58].

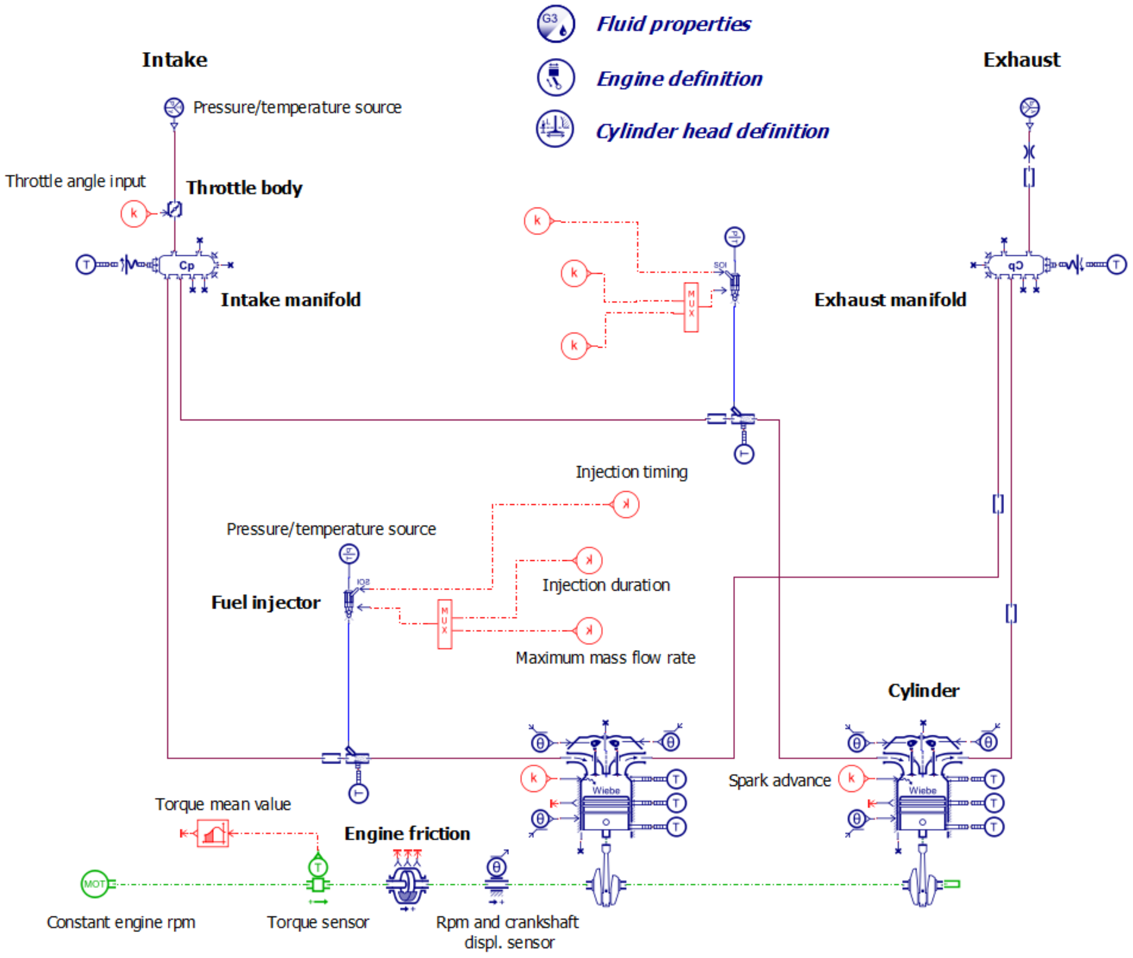


Figure 4.12: Auxiliary engine model.

it is now time to build the look-up table where the output is the pressure inside one of the engine's cylinders, and the inputs are the engine's crankshaft angular velocity (rpm) and crankshaft angular displacement. To achieve this, several simulations were performed setting different constant engine rpm values (from 1000 to 8500 rpm) and extracting the pressure inside cylinder 1 as function of the crankshaft angular displacement in a 720° range. For example, Fig. 4.13 represents the pressure inside cylinder 1 as function of the crankshaft angular displacement for a constant crankshaft angular velocity of 5000 rpm. After all data was collected and gathered, the look-up table was complete. The final result can be seen in Fig. 4.14.

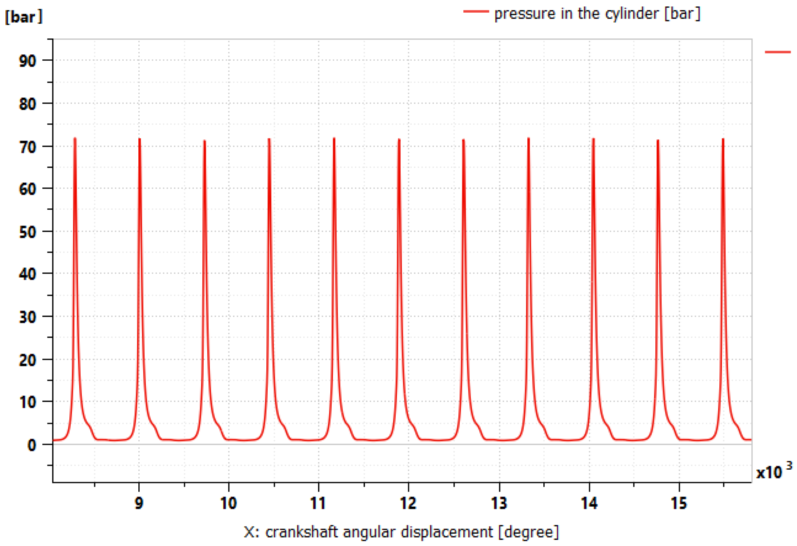


Figure 4.13: Cylinder pressure as function of crankshaft angular displacement for a constant crankshaft angular velocity of 5000 rpm.

4.2 Final engine model

The final engine model is characterized as a transient simulation in which the engine rotary velocity (rpm) can change throughout the same run, depending on a throttle command input. It will be centered around the look-up table created by the auxiliary engine model, presented in Fig. 4.14. In order to get the output engine torque it was also need to model the remaining parts of the motorcycle to obtain the feedback velocity input to the engine and a realistic engine load. In the future, when the model is connected with the real test bench only the engine part modeled should be considered.

4.2.1 Crankshaft-piston

The piston and crankshaft geometry and final torque calculations are similar to the ones explained in sections 4.1.1.3 and 4.1.1.4. However for this submodel the pressure inside the cylinder is not

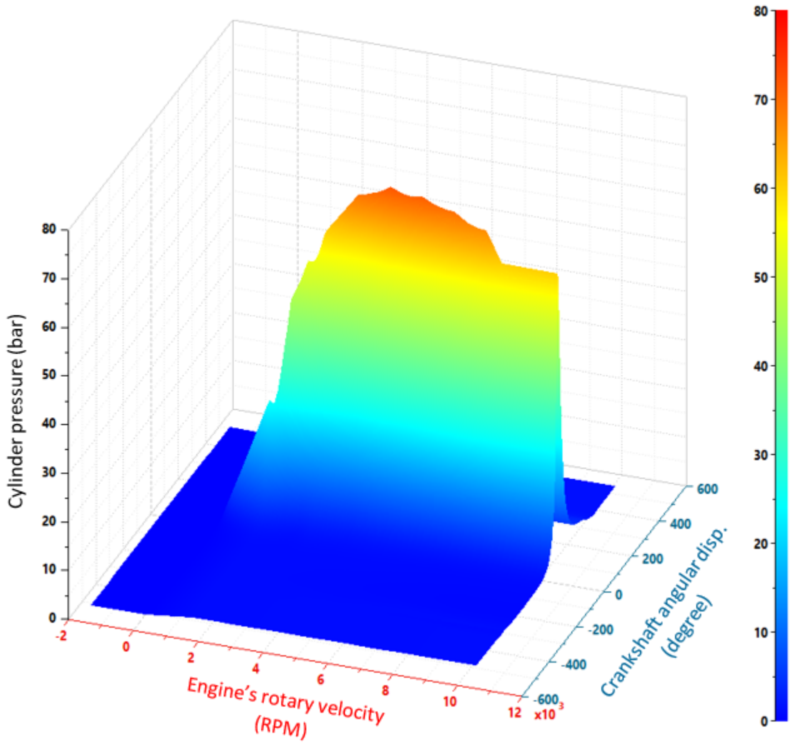


Figure 4.14: Cylinder pressure as function of engine rotary velocity and crankshaft angular displacement.

calculated as in section 4.1.1.2 but from a look-up table shown in Fig. 4.14. For each cylinder one crankshaft-piston submodel is needed as well as one look-up table. Important: there is a crankshaft angle displacement offset between the two crankshaft-piston submodels to replicate the pistons position offset inside the two cylinders. Furthermore, there is also a crankshaft angle displacement offset comparing the inputs entering in both lookup-tables in order to compute the output pressures with an offset as well. A special signal and control submodel is used to keep the crankshaft angular displacement input in the look-up tables between the values [-360°, 360°] (see Fig. 4.16). The referred offsets were added according to a 2010 Triumph Thunderbird model made by the 2020 *Simcenter Amesim* development team. Because this motorcycle has a four-stroke parallel twin engine and works as a 270° engine (one piston follows three quarters of a rotation behind the other) it can be said that yields identical firing intervals as a 90° V-twin engine, so the slightly "lumpy" feeling of this kind of engine is preserved [59].

4.2.2 Friction and inertia

Unlike the auxiliary engine model, this model takes into account an approximation of the global crankshaft's inertia as well as the flywheel's inertia. The use of the flywheel helps reduce the engine speed fluctuations, creating less noise in the speed sensor submodel used in the simulation model (see Fig. 4.16). To take into account these variables a rotary load submodel was added (see also Fig. 4.16). The rotary velocity is computed based on the load inertia, friction but also on the torque inputs on both ports. Friction is configurable and can be modeled as viscous friction, Coloumb friction, stiction and windage friction. So, the rotary velocity and angle are integrated with the following equation:

$$\frac{dw}{dt} = \begin{cases} (T - visc \cdot w - coul \cdot sign(w) - wind \cdot |w| \cdot w)\,/J, & \text{when } |T| > stict \text{ or } w \neq 0 \\ \\ 0, & \text{when } |T| < stict \text{ and } w = 0 \end{cases} \tag{4.34}$$

Where,

T = difference between torques at the 2 ports

$visc$ = viscous friction coefficient

w = load angular velocity

$coul$ = coulomb friction torque

$wind$ = coefficient of windage

$stict$ = stiction torque

J = moment of inertia

For the present case, friction wise, only viscous friction was implemented, using a viscous friction coefficient equal to 0.001 $\text{Nm} \cdot (\text{rev/min})^{-2}$, also used in 2010 Triumph Thunderbird model. The combined inertia value for the crankshaft and flywheel was also implemented in this submodel with a value equal to 0.0115 $\text{kg} \cdot \text{m}^2$, according to [60].

4.2.3 Starter

Because of the fact that the engine will start the simulation from standing position (zero initial velocity) it is necessary to add a starter motor and battery to initialize the engine model and make it start correctly. Besides, the look-up tables only give pressure output values from 1000 rpm, so without initial increase in crankshaft's rotary velocity to idle speed (around 1400 rpm according to Tab. 4.1) thanks to the effect of the starter motor, the simulation can't initiate properly. To make this initialization possible, 3 main components were modeled: battery, starter motor and starter gear. This construction was taken from a *Simcenter Amesim* demo model [19] where a starter was used. Variables and equations were not the concern here as the only purpose was to tune some parameters to achieve the desired initial increase in crankshaft speed. Figure 4.15 demonstrates the parallelism between the starter model and real life components. It is important to notice that the bigger and smaller pinion, which compose the starter gear, are not always in contact with each other. When the initial increase in rotary velocity happens a solenoid excitation occurs, pushing the smaller pinion in contact with the bigger one, connecting the starter motor shaft with the crankshaft and therefore the transmission occurs. To replicate this behavior, there is an input signal which determines when the starter motor shaft should engage with the crankshaft,

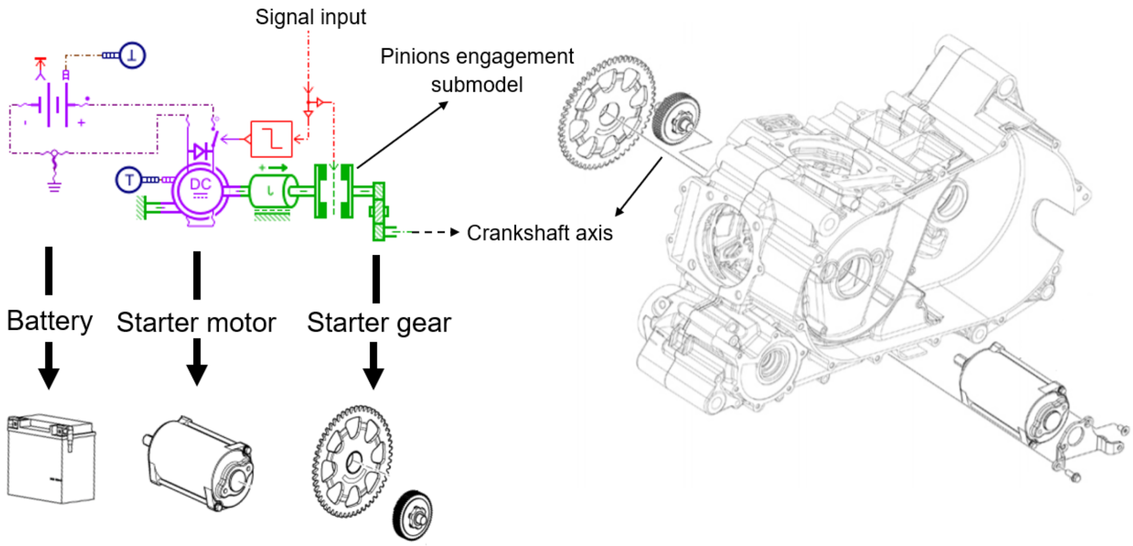


Figure 4.15: Starter model and its components.

also represented in Fig. 4.15. This signal, between 0 and 1, where 1 means fully engagement and 0 no engagement at all, is sent to a clutch submodel which works as the engagement between pinions submodel, transmitting movement depending on the input signal received.

4.2.4 Engine control

The previous mentioned submodels along with the engine control structure are shown in Fig. 4.16. The key visible in Fig. 4.16 has 3 different positions:

- **Value 0** — when engine is off
- **Value 1** — when engine is performing at normal operation conditions
- **Value 2** — when engine is starting, making the engine's starter work.

In this approach the engine load request works as a variable gain of the output pressure that comes from the look-up tables and goes to the crankshaft-piston submodel. This gain has a value between 0 and 1, determining how much pressure should enter in the combustion chamber. When the key value is equal 1 and the engine is performing at normal operation conditions two possible actions can take place:

- **No throttle applied** — only an idle speed request is applied making the engine rotary velocity go down to the idle speed and maintain that velocity until no different input is applied.
- **Throttle applied** — The throttle input also works as a gain, between 0 and 1, of the value 1 - indle speed request. The load request sent is equal to throttle request + idle speed request.

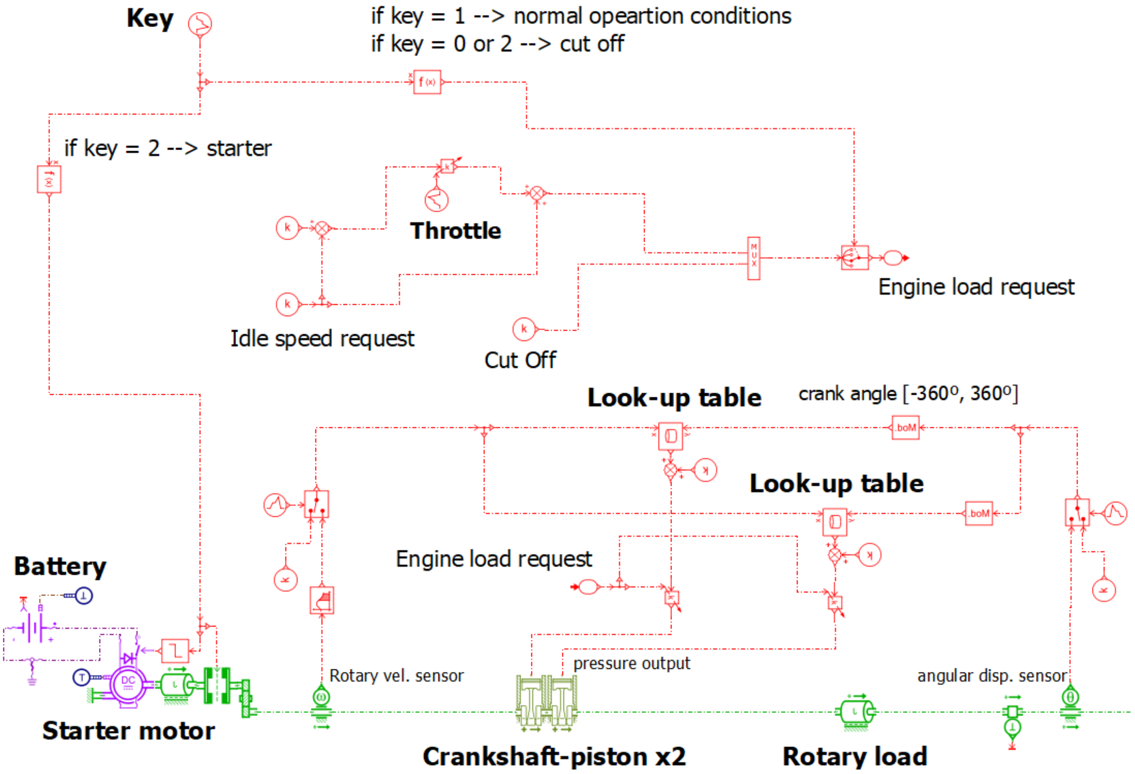


Figure 4.16: Final engine model and control structure.

4.2.5 eCVT

For the eCVT (electronically controlled Continuously Variable Transmission) a continuous variable transmission submodel was used. It is composed by two pulleys joined by a belt and each pulley can move axially to alter the operating radii of the belt on the pulleys and thus the transmission ratio. The submodel receives as inputs a value of torque and an axial force at each pulley. The axial forces are transformed into an axial displacement of the pulleys and in this way transmission ratio is changed. Transmission losses are not included and inertia of the pulleys is taken into account.

The rotary velocity of the driving pulley can be reached using Newton's second law of rotation and integrating the resulting equation 4.35.

$$\dot{w}_{driving} = \frac{\frac{T_{driven}}{ratio} - T_{driving} + \frac{J_{driven}}{ratio^3} \cdot \frac{d(ratio)}{dt} \cdot w_{driving}}{J_{driving} + \frac{J_{driven}}{ratio^2}} \tag{4.35}$$

As a result, the rotary velocity of the driven pulley can be represented as:

$$w_{driven} = \frac{-w_{driving}}{ratio} \tag{4.36}$$

Where,

$driving$	= with respect to the driving pulley
$driven$	= with respect to the driven pulley
w	= angular velocity
T	= torque input
$ratio$	= transmission ratio
J	= inertia

To calculate the transmission ratio and its evolution, presented in equation 4.37 (where R is the radial belt position), the following steps have to be done:

$$\frac{d(ratio)}{dt} = \frac{R_{driven}}{R_{driving}} = \frac{\dot{R}_{driven} \cdot R_{driving} - R_{driven} \cdot \dot{R}_{driving}}{{R_{driving}}^{2}} \tag{4.37}$$

Relate the axial forces applied in each pulley with the axial acceleration of the driving pulley, demonstrated in equation 4.38.

$$\dot{v}_{driving} = \frac{F_{driven} - F_{driving}}{M_{driving} + M_{driven}} \tag{4.38}$$

Where,

M	is the pulley mass,
F	is the axial force applied on the pulley,

The axial velocity of the driving pulley $v_{driving}$ is obtained by integration of equation 4.38. Next step is to relate the axial velocity of the driving pulley $v_{driving}$ with the belt radial velocity $\dot{R}_{driving}$, in the form of equation 4.39, where ε is half of the pulley v angle.

$$\dot{R}_{driving} = \frac{-v_{driving}}{2 \cdot tan(\varepsilon)} \tag{4.39}$$

Belt radial position $R_{driving}$ is obtained by integration of equation 4.39. The following step is to calculate the belt radial position R_{driven} and radial belt velocity $\dot{R}_{driven}$, with equations 4.40 and 4.41 respectively.

$$R_{driven} = R_{driving} + D \cdot sin(\varphi) \tag{4.40}$$

$$\dot{R}_{driven} = \dot{R}_{driving} + D \cdot cos(\varphi) \cdot \dot{\varphi} \tag{4.41}$$

Figure 4.17 ilustrates the variables used in the 2 previous equations. The final step is the computa-

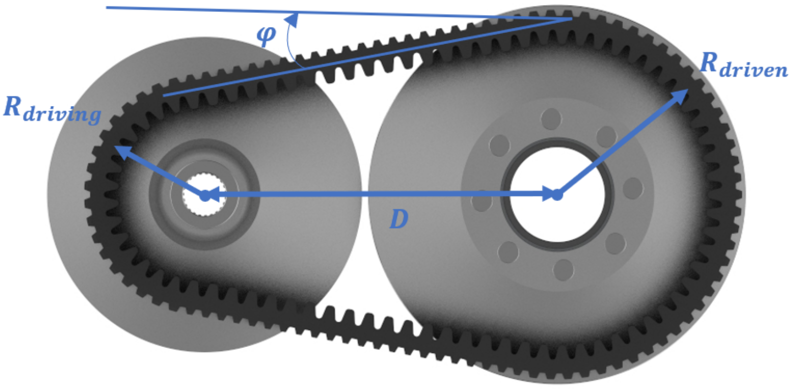


Figure 4.17: CVT belt side view [42].

tion of the φ angle and its derivative. To achieve that, relation 4.42 is needed, where L is the total constant belt length:

$$L = 2 \cdot D \cdot cos(\varphi) + \pi \cdot (R_{driving} + R_{driven}) + 2 \cdot \varphi \cdot (R_{driven} - R_{driving}) \tag{4.42}$$

To get a equation only dependent of φ and $R_{driving}$, equation 4.42 and equation 4.40 are combined into equation 4.43.

$$\left(\varphi + \frac{\pi}{2}\right) \cdot sin(\varphi) + cos(\varphi) = \frac{\pi}{D} \cdot \left(\frac{L}{2 \cdot \pi} - R_{driving}\right) \tag{4.43}$$

Equation 4.43 is derived into equation 4.44.

$$\dot{\varphi} = \frac{-\frac{\pi}{D} \cdot \dot{R}_{driving}}{(\varphi + \frac{\pi}{2}) \cdot cos(\varphi)} \tag{4.44}$$

Finally, all variables from equation 4.37 are defined and as a consequence the rotary velocity of the driving and driven pulleys can be calculated from equation 4.35 and equation 4.36. The submodel is initialized with an initial ratio and minimal radial belt position $R_{driving}$. For the present case there was no ratio control implemented, just a constant axial force applied in the driving pulley $F_{driving}$ resulting in a constant ratio throughout the whole simulation, equal to 1.2685. On the driven pulley a constant opposing force F_{driven} is generated by a spring-damper submodel.

All variables needed regarding the eCVT (masses, inertias and geometry) were taken from [42].

4.2.6 Centrifugal clutch

As stated in section 3.1, motorcycles with CVT transmission type commonly have a centrifugal clutch. Aprilia Mana 850 GT's centrifugal clutch is composed by a spider, multiple shoes, springs and a drum. A schematic representation of the actuall clutch and its behavior is made in Fig. 4.18. A shaft is attached to the spider which is connected to the lower far end of each shoe. With increasing shaft's angular velocity the centrifugal clutch's shoes move radially thanks to the centrifugal forces, until the friction blocks come in full contact with the drum. As a consequence, the shaft attached to the spider transmits movement to the shaft attached to the drum [61]. The centrifugal force applied on one shoe is represented in equation 4.45, also called the D'Alembert force caused by the centripetal acceleration of a rotation body.

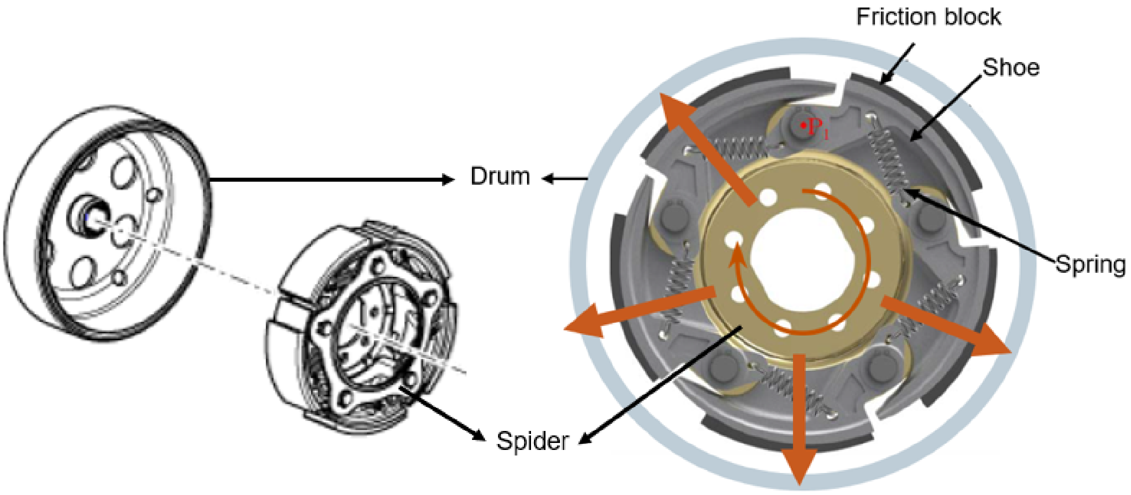


Figure 4.18: Centrifugal clutch components.

$$F_{centrifugal} = m \cdot a = m \cdot r_{shoe} \cdot w^2 \tag{4.45}$$

Where,

m = shoe mass

w = spider angular velocity

r_{shoe} = distance from spider center to the shoe

The springs attached to the shoes make the clutch engage at a determined speed (which should be slightly above 1400 rpm, the idle speed) by creating an opposing force to the centrifugal one. For simplification purposes and to avoid doing a free body diagram on one shoe like in [61] (case with clutch with just two shoes), the springs force is related with the spider's shaft angular velocity threshold, w_0, for shoes and drum engagement. The normal force acting on the shoe is presented in equation 4.46 [62].

$$F_{normal} = \begin{cases} F_{centrifugal} - F_{\text{shoe springs}} = m \cdot r_{shoe} \cdot w^2 - m \cdot r_{shoe} \cdot {w_0}^2, & \text{if } w > w_0 \\ \\ 0, & \text{if } w \leq w_0 \end{cases} \tag{4.46}$$

The friction torque generated between the drum and the shoes will be equal to equation 4.47.

$$T_{friction} = N \cdot F_{normal} \cdot \mu \cdot r_{drum} \tag{4.47}$$

Where,

N	= number of shoes
μ	= friction coefficient between the drum and shoe
r_{drum}	= drum inside radius

To model the above equations in *Simcenter Amesim* the approach implemented in [63], where a two-wheeler powertrain is simulated, was adopted. A clutch submodel was applied receiving an input normal force and computing the output friction torque between two rotating bodies, based on the friction coefficient and diameter on which friction acts. The normal force input is calculated with a function block representing equation 4.46 where the input is the current crankshaft rpm taken from a rotary velocity sensor.

The variables used in this section have their values represented in Tab. 4.2.

Table 4.2: Clutch variables

Variable	**Value**
Shoe mass (m)	0.1 kg
Distance from spider center to the shoe (r_{shoe})	39 mm
Number of shoes (N)	5
Drum inside radius (r_{drum})	108 mm
Friction coefficient between the drum and shoe (μ)	0.3

4.2.7 Gear drive and chain drive

The first transmission after the centrifugal clutch is a gear drive between the output shaft (coupled to the centrifugal clutch's drum) gear and a second shaft gear. A parallelism between the real transmission and the *Simcenter Amesim* submodels is shown in Fig. 4.19.

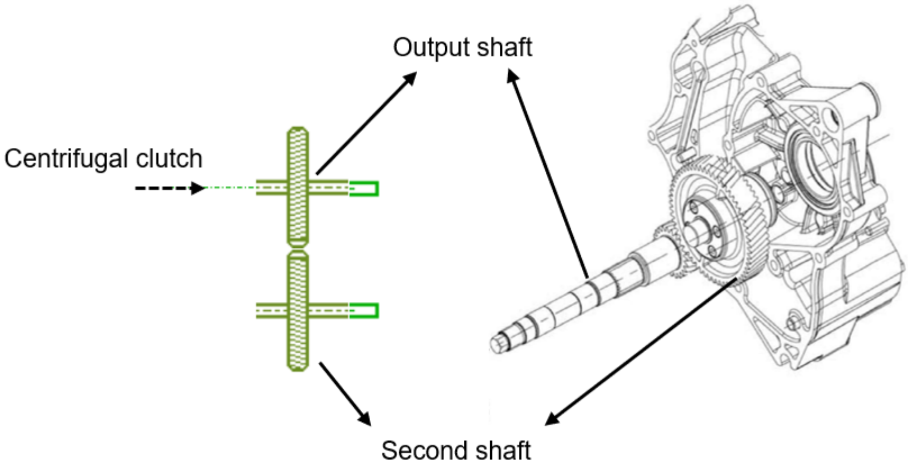


Figure 4.19: First transmission after the centrifugal clutch.

The second transmission is a chain drive that connects the second shaft sprocket with the rear wheel sprocket. A parallelism between the real transmission and the *Simcenter Amesim* submodels is shown in Fig. 4.20.

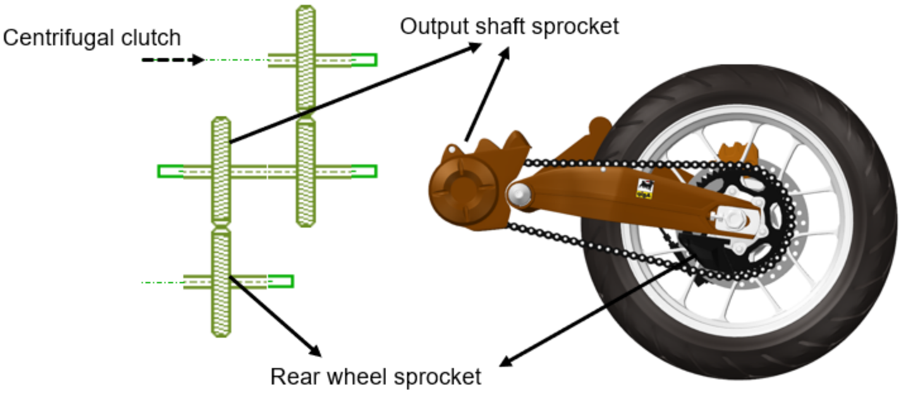


Figure 4.20: Second transmission - chain drive.

For both transmissions, the torque on the driving gear and the torque on the driven gear can be related as in equation 4.48. R_1 and R_2 are the number of teeth of the driving gear and driven gear respectively.

$$T_{driven} = \frac{T_{driving} \cdot R_2}{R_1} = \frac{T_{driving}}{\text{gear ratio}} \tag{4.48}$$

For both transmissions, the angular velocity on the driving gear and the angular velocity on the driven gear can be related as in equation 4.49.

$$w_{driven} = \frac{w_{driving} \cdot R_1}{R_2} = w_{driving} \cdot \text{gear ratio} \tag{4.49}$$

According to [45], the gear ratio of the gear drive is equal to 23/59 and the gear ratio of the chain drive is equal to 18/40.

4.2.8 Motorcycle's body, tires, road and model overview

To model the motorcycle's body, tires and road it was applied the approach used in [42]. Equations behind this submodels won't be specified because these parts were also simulated in detail in chapter 5 resorting *Simcenter 3D Motion*. In few words the motorcycle's body submodel uses a 2D model with 3 degrees of freedom: pitch rotation, longitudinal and vertical displacement. It is possible to obtain the position, velocity and acceleration of the body's center of gravity (COG). To compute this values it is necessary to receive as input forces generated in motorcycle's suspensions and tires submodels. A braking torque input in the front and rear wheel is also available. Additionally, the motorcycle is running on a flat road. Figure 4.21 shows the model created for other parts except the engine. Figure 4.22 shows the connection between the final engine model with the previous mentioned model.

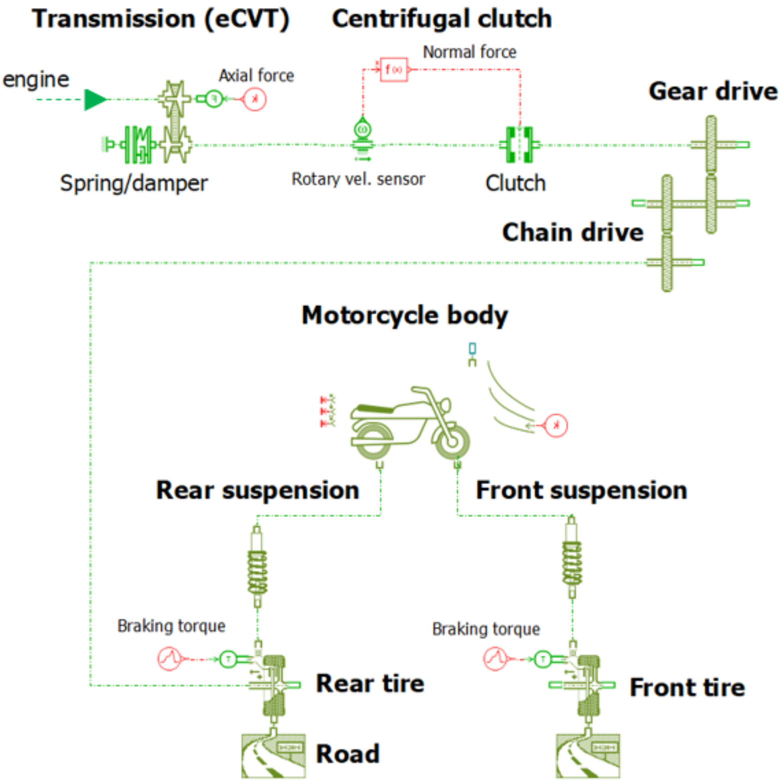


Figure 4.21: Other motorcycle parts model for speed feedback and realistic engine load.

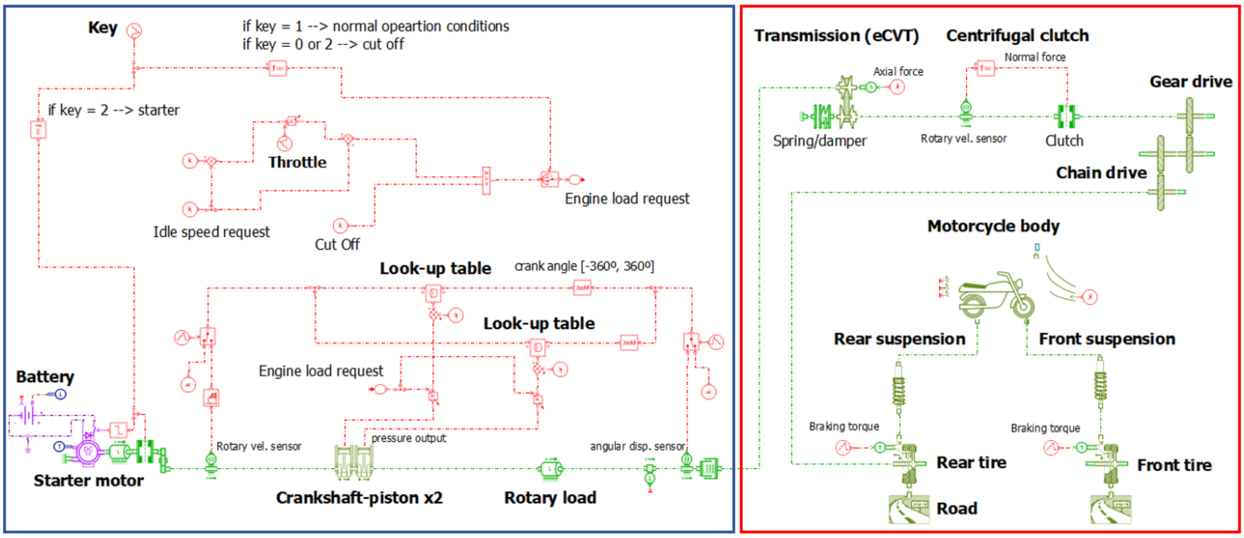


Figure 4.22: Motorcycle model: Inside blue square - final engine model; Inside red square - other motorcycle parts for speed feedback and realistic engine load.

Chapter 5

Motorcycle multibody modeling

As previously mentioned in chapter 3, the work of this dissertation is centered around Aprilia Mana 850 GT motorcycle. The multibody model was the second to be done, resorting to the modeling tool *Simcenter 3D Motion*, already introduced in section 2.5.1, in order to obtain the motorcycle dynamics and connect it to the eCVT, according to Fig. 3.7.

In multibody dynamics formulation and from the kinematic point of view, every mechanical system consists of a number of rigid bodies connected to each other by a number of joints. Each body has six degrees of freedom (DOF) since its position and orientation in space are fully defined by six parameters, such as three coordinates of a point (x, y, z) and three angles (yaw, roll, pitch). When a joint is included, the number of DOFs reduces according to the type of joint used [64]. The same approach can be applied to a motorcycle being normally divided in six rigid bodies [65]:

1. Front wheel composed by the front tire and rim.
2. Rear wheel composed by the rear tire and rim.
3. Rear frame including the driver (simpler case), the engine, chassis and fuel mass.
4. The steering head with handlebar and the suspended part of the front fork.
5. The unsuspended part of the front fork.
6. Rear swing arm.

The bodies' dimensions and forms were based on IGES files bought for motorcycle 3D visualization purposes. Table 5.1 shows some general dimensions from the motorcycle and its mass.

5.1 Bodies' modeling

First task at hands was to transform the motorcycle parts presented in the IGES files from surfaces to solid bodies resorting *SolidWorks*. Thanks to this it was possible to assign materials to the

Table 5.1: Motorcycle general dimensions and mass [45].

Specification	**Quantity**
Max. length	2180 mm
Max. width	800 mm
Max. height	1130 mm
Wheelbase	1470 mm
Kerb weight (full fuel tank)	230 kg

different bodies and obtain centers of mass, inertia and mass properties. The bodies created are shown in Fig. 5.1 and enumerated according to the introductory section of this chapter.

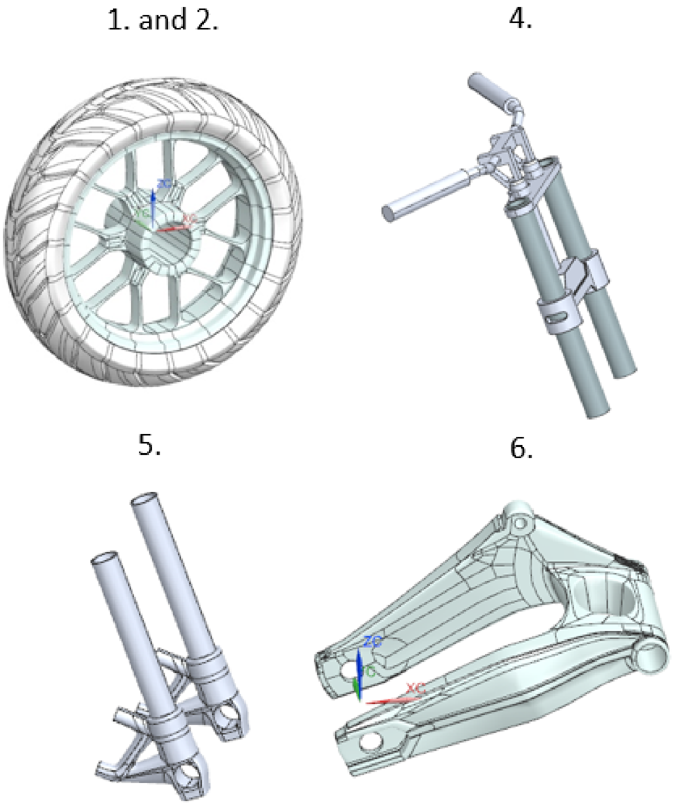


Figure 5.1: Motorcycle parts created.

For the rear frame, it was not possible to form solid bodies from IGES files surfaces. To surpass this set back it was made a chassis simplification with the help of 3D sketches in *Siemens NX*. This feature was also used to create 2 dummy shafts to replicate the driving shaft and the driven shaft of the gear drive (already explained in section 4.2.7). Figure 5.2 illustrates the assembled motorcycle and the 2 dummy shafts created.

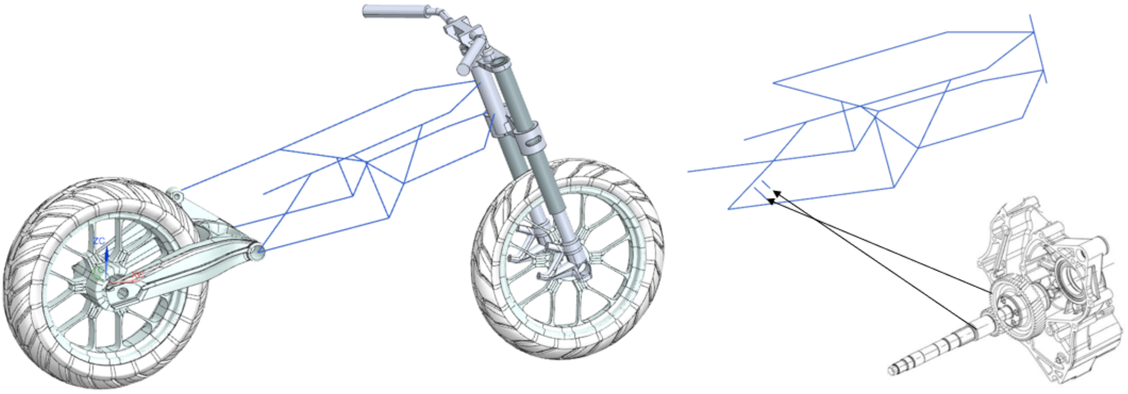


Figure 5.2: Motorcycle assembly (left) and 2 dummy shafts (right).

5.2 Motion bodies

Following the motorcycle parts creation and assembly, the multibody model started to be built through the use of *Simcenter 3D Motion*.

First step of model creation was to associate the different rigid bodies built in section 5.1 to motion bodies. In order to do that, for each motion body the center of mass, mass and inertia properties had to be defined. For the solid bodies transformed from the IGES files, the center of mass, mass and inertia properties were calculated automatically by the software by setting the material of the part.

For the rear frame (associated with chassis simplification geometry) the center of gravity was taken from a Ducaty road type motorcycle with the same wheelbase as the Aprilia Mana 850 GT [66]. For the rear frame mass calculation it was removed from the motorcycle kerb weight (230 kg) plus the rider (70 kg) the mass of all other solid bodies created. The inertia properties were copied from a motorcycle with similar Aprilia Mana 850 GT's rear frame mass [65].

Table 5.2 shows the center of mass, mass and inertia properties of all motion bodies defined. Important to notice that x axis refers to motorcycle forward movement direction, z axis to the vertical direction pointing to the ground and the remaining y axis pointing according to the right hand rule. The center of gravity is defined relative to the rear wheel axis of revolution. For the dummy shafts, a really low mass and low inertia values were defined. Finally, a motion body was also needed for road creation in a section ahead.

5.3 Joints

Motorcycle models usually have 11 degrees of freedom and can be represented by the vector $q^T = (x, y, z, \psi, \chi, \phi, \chi_{sw}, \chi_r, \beta, s_f, \chi_f)^T$. Figure 5.3 identifies the previous vector variables. For this dissertation, the following simplifications were made: the motorcycle will only run in a straight line along its middle plane which means that y, ψ, ϕ and β will be constrained to 0, leaving the model with 7 degrees of freedom. The relative motion between bodies was defined

Table 5.2: Motion bodies properties

Motion body	**Mass** [kg]	**Center of gravity (x,y,z)** [mm]	**Inertia** (I_{xx}, I_{yy}, I_{zz}; I_{xy}, I_{yz}, I_{zx}) [kg · m^2]
Rear rim	7.41	(0,0,0)	(0.131, 0.079, 0.079, 0, 0, 0)
Rear tire	7.76	(0,0,0)	(0.642, 0.340, 0.340, 0, 0, 0)
Front rim	3.49	(1455, 0, 4.75)	(0.081, 0.042, 0.042, 0, 0, 0)
Front tire	4.48	(1455, 0, 4.75)	(0.339, 0.173, 0.173, 0, 0, 0)
Rear swing arm	11.91	(246.27, 0, 95.67)	(0.331, 0.185, 0.175, 0, 0, 0)
Rear frame	250	(655, 0, 285.10)	(16.582, 34.758, 23.632, 0, 0, 0)
Susp. part front fork	12.99	(1217.41, 0, 474.46)	(0.646, 0.425, 0.250, 0, 0, 0)
Unsusp. part front fork	6.66	(1406.27, 0, 53.29)	(0.133, 0.094, 0.052, 0, 0, 0)
Driving shaft (dummy)	0.5	(566.85, 12.57, 123.54)	(0.0001, 0.0001, 0.0001, 0, 0, 0)
Driven shaft (dummy)	0.5	(566.85, 12.57, 123.54)	(0.0001, 0.0001, 0.0001, 0, 0, 0)

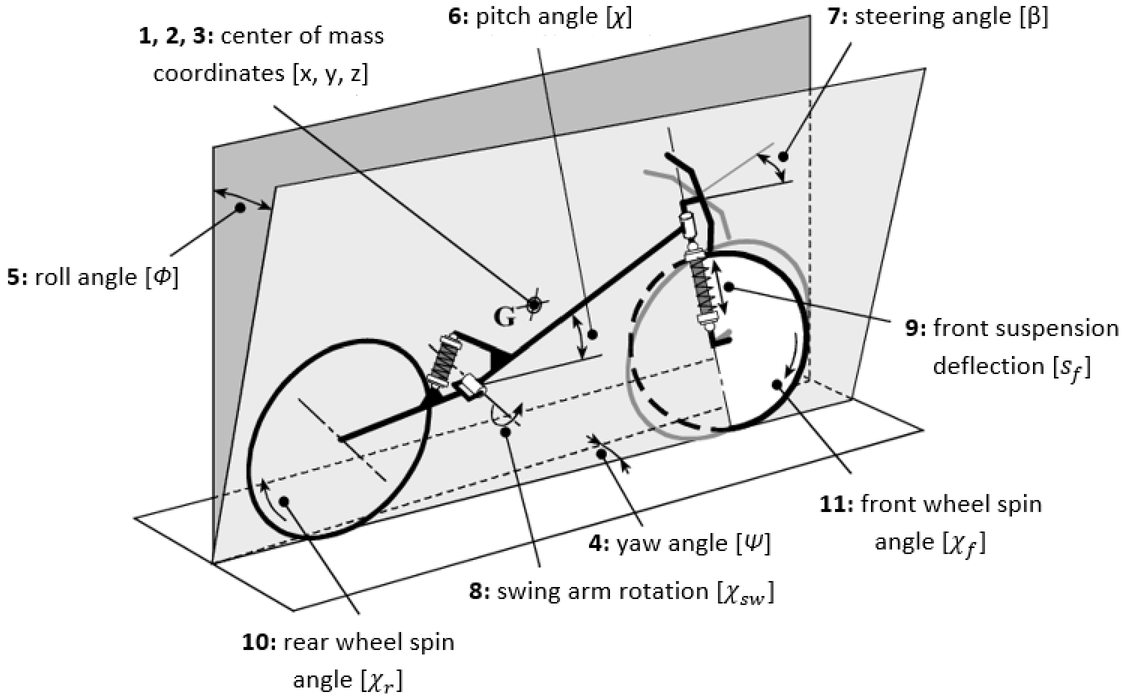

1, 2, 3: center of mass coordinates [x, y, z]
6: pitch angle [χ]
7: steering angle [β]
5: roll angle [Φ]
G
9: front suspension deflection [s_f]
11: front wheel spin angle [χ_f]
4: yaw angle [Ψ]
8: swing arm rotation [χ_{sw}]
10: rear wheel spin angle [χ_r]

Figure 5.3: Motorcycle model with 11 degrees of freedom [67].

(applying a total of 10 joints) to give the motorcycle 7 degrees of freedom. Figure 5.4 specifies the joints used for the model construction. Locking the steering wheel rotation (shown in section 5.6) and setting a chain drive and gear drive (shown in section 5.5) is also needed to achieve the final 7 degrees of freedom.

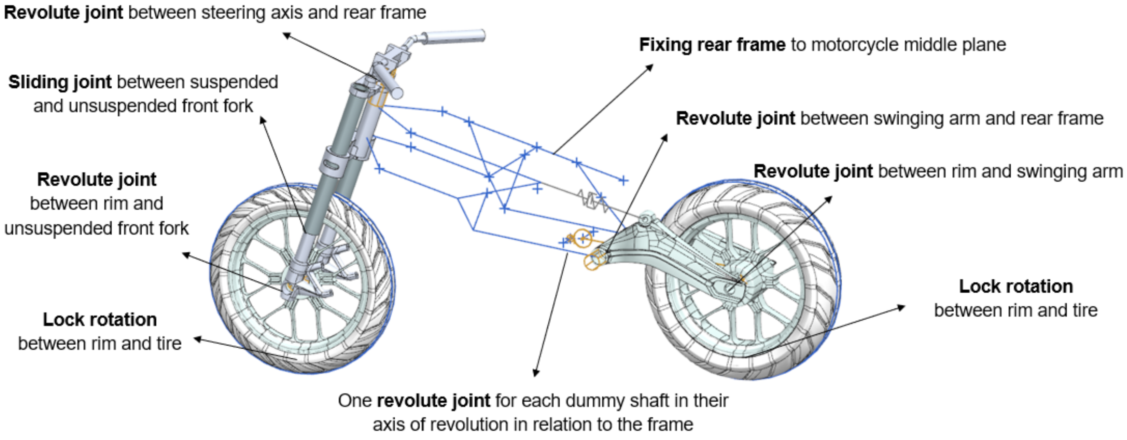

Revolute joint between steering axis and rear frame
Sliding joint between suspended and unsuspended front fork
Revolute joint between rim and unsuspended front fork
Lock rotation between rim and tire
Fixing rear frame to motorcycle middle plane
Revolute joint between swinging arm and rear frame
Revolute joint between rim and swinging arm
Lock rotation between rim and tire
One **revolute joint** for each dummy shaft in their axis of revolution in relation to the frame

Figure 5.4: Motorcycle joints.

5.4 Suspensions

The Aprilia Mana 850 GT has the most widespread front suspension which is, undoubtedly, the telescopic fork. It is made up of two telescopic sliders which run along the inner tube of the fork and form a slider joint between the suspended and unsuspended mass of the front motorcycle frame. For this particular case, the telescopic fork is not in a classic disposition but upside-down.

The rear suspension is a variant from the classic rear suspension and is called cantilever monoshock, characterized by only one spring-damper unit. The spring-damper unit is attached between the swing arm and rear frame [68].

For the front suspension a spring-damper unit was created between two points of two motion bodies (suspended part of the front fork and unsuspended part of the front fork). These points can be seen in Fig. 5.5 and are separated by an installed length of 250 mm. The spring's free length L_0 is calculated by equation 5.1.

$$L_0 = \frac{F}{k} + L \tag{5.1}$$

Where,

F	= preload force on the spring
k	= spring's stiffness
L	= installed length

As the spring's stiffness was set to 18 kN/m (normal values for front suspension are in the range 13-25 kN/m [64]) and the spring's free length is 300.8 mm according to [50], a 914 N preload force on the spring has to be applied. The value for the damping coefficient was set to 550 $\mathrm{N \cdot s/m}$ (normal values for front suspension are in the range 500-2000 $\mathrm{N \cdot s/m}$).

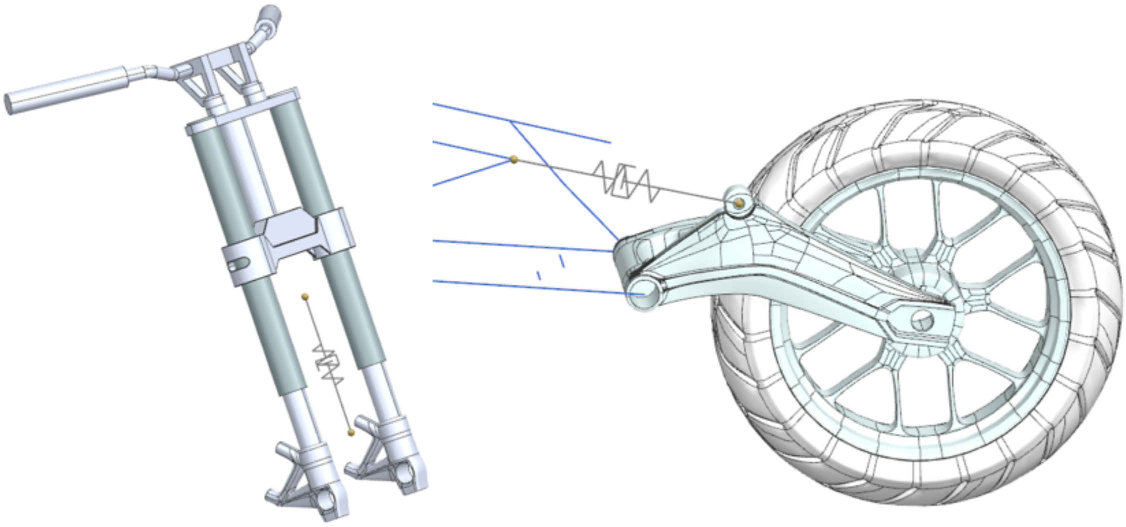


Figure 5.5: On the left: front suspension; on the right: rear suspension.

For the rear suspension, equation 5.1 can be applied and the two points, set between the swing arm and rear frame, are separated by an installed length of 351.95 mm. The spring's stiffness was set to 140 kN/m (normal values for front suspension are in the range 100-150 kN/m [64]) and a 500 N preload force was set to keep the motorcycle in equilibrium position. The value for the damping coefficient was set to 5 $\mathrm{kN \cdot s/m}$ (normal values for front suspension are in the range 5-15 $\mathrm{kN \cdot s/m}$).

5.5 Gear drive and chain drive

To replicate the real gear drive between shafts, already shown in section 4.2.7, more specifically in Fig. 4.19, and the chain drive, shown in Fig. 4.20, was used the *Simcenter 3D Motion* option “gear drive” and the option “2 joint coupler”.

Thanks to the gear drive, 2 revolute joints created for each dummy shaft about their axis of revolution (see Fig. 5.4) are now connected by setting their gear ratio equal to 23/59. Figure 5.6 shows the model's gear drive.

Thanks to the chain drive, the revolute joint of the driven shaft of the gear drive and the revolute joint between the rim and swing arm are now connected by setting their gear ratio equal to 18/40. Figure 5.7 shows the chain drive model. With this design, the input torque of the model coming from the eCVT test bench (see Fig. 3.7) will be sent directly to the driving shaft (dummy shaft)

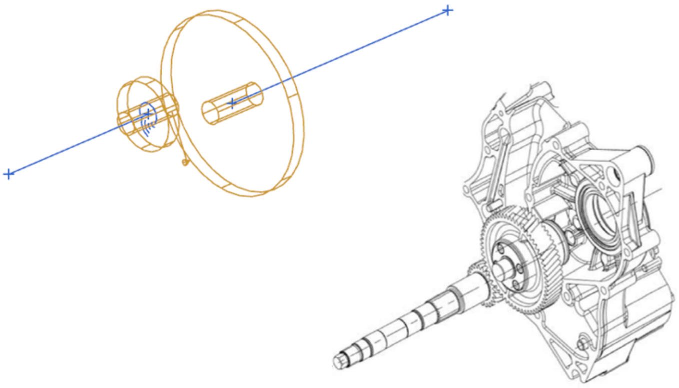


Figure 5.6: Motorcycle's gear drive

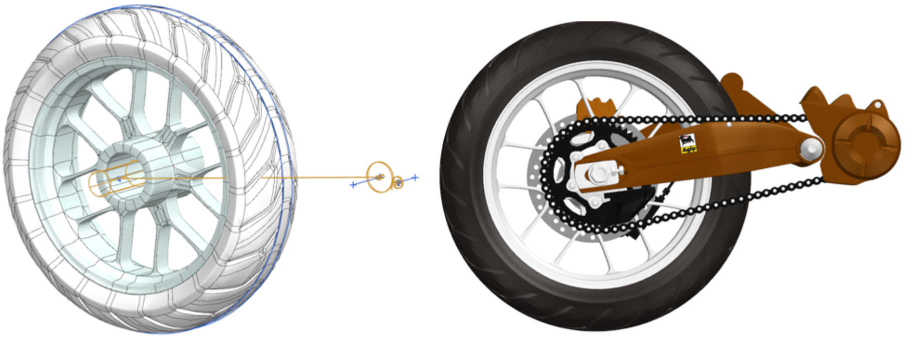


Figure 5.7: Motorcycle's chain drive.

of the gear drive and the movement will be transmitted to the rear wheel with the help of the gear and chain drive created.

5.6 Loads and drivers

In order to keep the motorcycle moving on a straight line, even knowing that is an unstable system, locking the steering wheel was necessary. To do that it was applied a driver to the revolute joint between the steering axis and the rear frame (see Fig. 5.4) constraining its angular velocity to 0. While the co-simulation isn't fully operational a torque load was applied on the driving shaft of the gear drive to study the motorcycle model behavior. It is possible to set different load profiles with respect to time.

For this dissertation was applied a constant value of 70 $\mathrm{N \cdot m}$ torque throughout the whole simulation. Figure 5.8 demonstrates the torque load and driver used on this model.

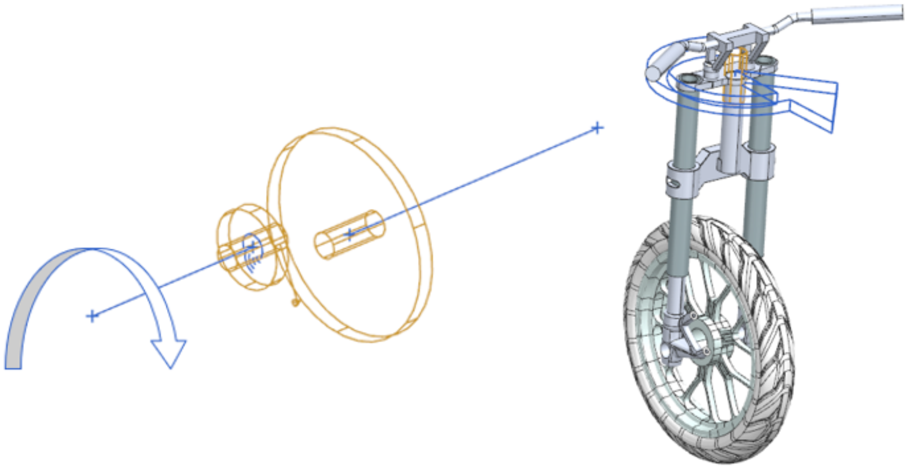


Figure 5.8: On the left: Torque load input; on the right: Steering wheel lock.

5.7 Tire parameters

Since the advent of the first pneumatic tire that it has gradually become one of the motorcycle's most important component. The tire characteristics have a crucial role to play in the motorcycle performance. Tire simulation is then very complicated but also the most important part to achieve an accurate multibody model [69]. This complex system can be roughly modeled as three forces and three moments, which are described in Fig. 5.9 and enumerated bellow:

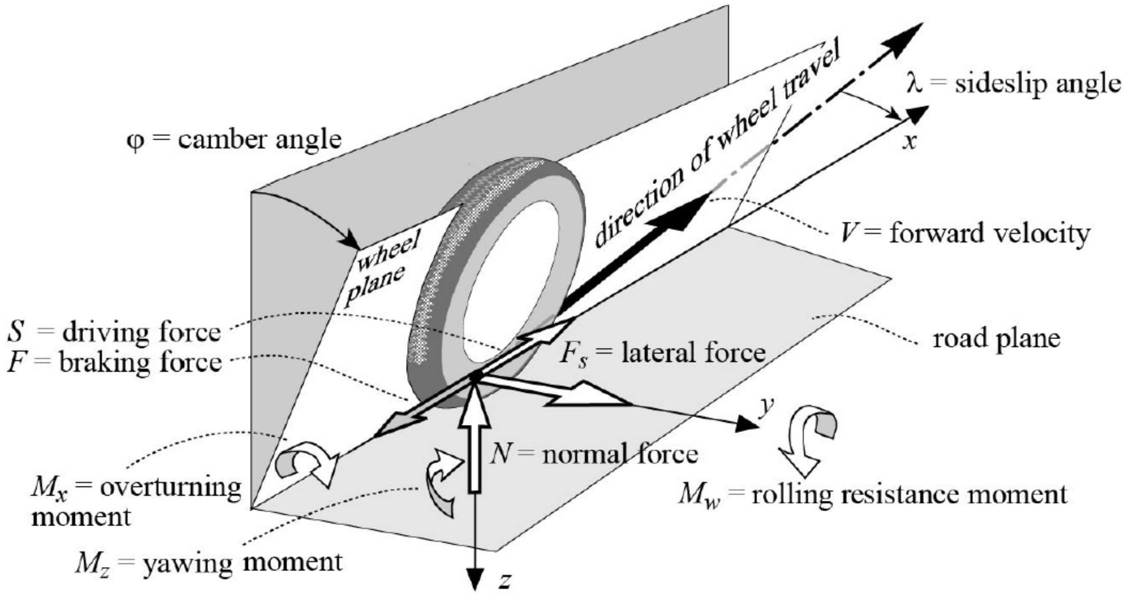


Figure 5.9: Total of forces present in a motorcycle tire [68].

- **Longitudinal force** — force acting along the axis parallel to the intersection of the wheel plane with the road plane, and passing through the contact point (assumed positive if driving and negative if braking), in x direction.

- **Vertical force** — force orthogonal to the road plane (a vertical load that acts on the wheel, assumed positive in an upward direction), along the z axis.
- **Lateral force** — force in the road plane, orthogonal to the longitudinal force, in y direction.
- **Overturning moment** — moment around the x-axis.
- **Rolling resistance moment** — moment around the y -axis.
- **Yawing moment** — moment around the z-axis.

To calculate steady-state tire forces and moments for motorcycle dynamic analysis the *Magic formula* was used. The *Magic formula* model, first proposed by Pacejka in 1993, is widely used in both automotive and motorcycle tire modeling. The functions were developed from a purely empirical approach, and were made to relate normal tire force, longitudinal slip, sideslip angle and camber angle to longitudinal force, lateral force and aligning moment (shown in Fig. 5.9). Trough continuous exploration and evolution, the *Magic Formula* has evolved into a single expression that can represent the lateral or longitudinal force or aligning moment as below [70, 69]:

$$y(x) = D \cdot sin\left[C \cdot arctan\left(B \cdot x - E \cdot (B \cdot x - arctan(B \cdot x))\right)\right] \tag{5.2}$$

With,

$$Y(X) = y(x) + S_v \tag{5.3}$$

$$x = X + S_H \tag{5.4}$$

Where,

B, C, D, E = coefficients dependent on the tire normal load and camber angle
S_V = horizontal offset of the curve along the x axis
S_H = vertical offset of the curve along the y axis
x = either the tire sideslip angle λ or the tire longitudinal slip κ
y = either the longitudinal force, lateral force or aligning moment

Only a complete set of parameter values (that defines B, C, D and E) for a given tire will allow the calculation of the steady-state force and moment system for any realistic operating condition, thus its difficult to accurately represent them as there is a need of experimental data on the particular tire and a very wide range of different motorcycle tires exist.

For this model the motorcycle goes on a straight line along its middle plane in what is called a pure longitudinal slip. In this case the sideslip angle λ and camber angle φ are set to zero so that the tire does not generate any lateral force, yawing moment and overturning moment, leaving only a need to calculate the tire longitudinal force. From [71], ignoring and omitting certain offset terms from the *Magic formula* relations, in order to model the generic rather than the particular, expression for pure longitudinal slip case is shown in the following equations:

$$df_z = (F_z - F_{z0})/F_{z0} \tag{5.5}$$

$$F_{long} = D \cdot sin\left[C \cdot arctan\left(B \cdot \kappa - E \cdot (B \cdot \kappa - arctan(B \cdot \kappa))\right)\right] \tag{5.6}$$

$$D = (p_{D1} + p_{D2} \cdot df_z) \cdot F_Z \tag{5.7}$$

$$E = (p_{E1} + p_{E2} \cdot df_z + p_{E3} \cdot df_z{}^2) \cdot (1 - p_{E4} \cdot sgn(\kappa)) \tag{5.8}$$

$$K_\kappa = F_z \cdot (p_{K1} + p_{K2} \cdot df_z) \cdot e^{p_{K3} \cdot df_z} \tag{5.9}$$

$$B = \frac{K_\kappa}{C \cdot D} \tag{5.10}$$

Where,

df_z = normalized change in tire normal force (non-dimensional increment)

F_z = tire normal force

F_{z0} = reference vertical force, nominal force equal to 1600 N

Specific data for micro-coefficients shown in the above equations for B, C, D and E coefficients calculation is needed for Aprilia Mana 850 GT's front and rear tire. The tire nomenclature is 120/70 ZR"17 for the front one and 180/55 ZR17" for the rear one. Micro-coefficients values where taken from [72]. Data for the 120/70 ZR"17 front tire and 180/55 ZR"17 rear tire are presented in Tab. 5.3.

Table 5.3: Micro-coefficients for 120/70 ZR"17 front tire and 180/55 ZR"17 rear tire for pure longitudinal slip [72].

Tire	C	p_{D1}	p_{D2}	p_{E1}	p_{E2}	p_{E3}	p_{E4}	p_{K1}	p_{K2}	p_{K3}
120/70 ZR"17	1.6064	1.381	-0.04143	0.0263	0.27056	-0.0769	1.1268	25.94	-4.233	0.3369
180/55 ZR"17	1.6064	1.381	-0.04143	0.0263	0.27056	-0.0769	1.1268	25.94	-4.233	0.3369

A problem arise because the software program routine for magic formula coefficients calculation uses equations from [70], dated 1992, instead of using equations shown above from [71], from 2006, in which the tire data gathered is based on. To keep data valid the equations 5.6-5.10 needed to be manipulated in order to be possible to introduce the final micro-coefficients derived from the ones in Tab. 5.3 into the program subroutine (Fortran file). The main difference between the *Magic formula* equations between [70] (from 1992) and [71] (from 2006) is that the second one uses a normalized change in tire normal force (a non-dimensional increment) df_z, and the first one uses only the tire normal force to compute the coefficients B, C, D and E. The program subroutine is defined by a vector with 14 coefficients (represented as q), which is implemented in the model in the form of a txt file. The manipulation of equations 5.6-5.10 to obtain the vector q

14 coefficient values from the coefficients shown in Tab. 5.3, is represented below:

For coefficient D calculation

$$D_{\text{2004 equations}} = (p_{D1} + p_{D2} \cdot df_z) \cdot F_Z \tag{5.11}$$

$$D_{\text{2004 equations}} = (p_{D1} - p_{D2} + \frac{p_{D2}}{F_{z0}} \cdot F_z) \cdot F_Z \tag{5.12}$$

$$D_{\text{program routine}} = (q(2) + q(1) \cdot F_z) \cdot F_z \tag{5.13}$$

Which gives,

$$q(1) = \frac{p_{D2}}{F_{z0}} \tag{5.14}$$

$$q(2) = p_{D1} - p_{D2} \tag{5.15}$$

For coefficient E calculation

$$E_{\text{2004 equations}} = (p_{E1} + p_{E2} \cdot df_z + p_{E3} \cdot {df_z}^2) \cdot (1 - p_{E4} \cdot sgn(\kappa)) = \tag{5.16}$$

$$= (p_{E1} - p_{E2} + p_{E3} + (\tfrac{p_{E2}}{F_{z0}} - \tfrac{2 \cdot p_{E3}}{F_{z0}}) \cdot F_z + \tfrac{p_{E3}}{F_{z0}^2} \cdot {F_z}^2) \cdot (1 - p_{E4} \cdot sgn(\kappa)) \tag{5.17}$$

$$E_{\text{program routine}} = (q(8) + q(7) \cdot F_z + q(6) \cdot {F_z}^2) \cdot (1 - q(13) \cdot sgn(\kappa)) \tag{5.18}$$

Which gives,

$$q(6) = \frac{p_{E3}}{F_{z0}^2} \tag{5.19}$$

$$q(7) = \frac{p_{E2}}{F_{z0}} - \frac{2 \cdot p_{E3}}{F_{z0}} \tag{5.20}$$

$$q(8) = p_{E1} - p_{E2} + p_{E3} \tag{5.21}$$

$$q(13) = p_{E4} \tag{5.22}$$

For coefficient K_κ calculation

$$K_{\text{2004 equations}} = F_z \cdot (p_{K1} + p_{K2} \cdot df_z) \cdot e^{p_{K3} \cdot df_z} \tag{5.23}$$

$$K_{\text{2004 equations}} = ((p_{K1} - p_{K2}) \cdot F_z + \frac{p_{K2}}{F_{z0}} \cdot {F_z}^2) \cdot \frac{e^{p_{K3} \cdot \frac{F_z}{F_{z0}}}}{e^{p_{K3}}} \tag{5.24}$$

$$K_{\text{program routine}} = (q(3) \cdot {F_z}^2 + q(4) \cdot F_z) \cdot e^{-q(5) \cdot F_z} \tag{5.25}$$

Which gives,

$$q(3) = \frac{p_{K2}}{F_{z0} \cdot e^{p_{K3}}} \tag{5.26}$$

$$q(4) = \frac{p_{K1} - p_{K2}}{e^{p_{K3}}} \tag{5.27}$$

$$q(5) = \frac{e^{p_{K3}}}{F_{z0}} \tag{5.28}$$

Finally, q(0) = C and q(9), q(10), q(11) and q(12) are all equal to 0 in order to make offset terms S_V and S_H, presented in equations 5.3 and 5.4, equal to 0, modelling a generic 120/70 ZR”17 front tire and 180/55 ZR"17 rear tire rather than the particular ones used in this motorcycle (no data available for those).

To get the coefficients B, D and E for longitudinal force calculation, besides the vector q already defined it is also needed to obtain the tire normal force F_z. In order to do that, the software computes the tire deflection based on the wheel center position, wheel radius and ground position. The wheel radius for a 120/70 ZR"17 front tire is 299 mm and for a 180/55 ZR"17 rear tire is 314.9 mm. Tire deflection is shown in Fig. 5.10. Finally, tire normal force calculation is obtained with equation 5.29.

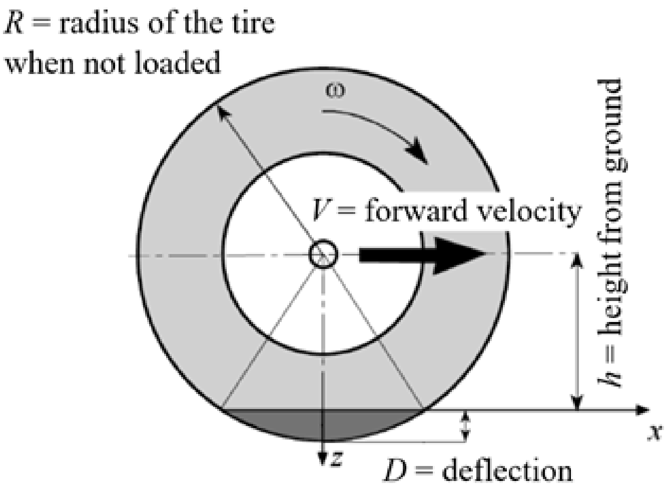


Figure 5.10: Tire deflection. Adapted from [68].

$$F_z = k_{vertical} \cdot D - c_{vertical} \cdot v_z \tag{5.29}$$

Where,

$k_{vertical}$ = tire vertical stiffness

$c_{vertical}$ = tire vertical damping

v_z = tire normal velocity

Front tire vertical stiffness $k_{vertical}$ and damping $c_{vertical}$ were set to 25000 N/m and 2134 $\mathrm{N \cdot s/m}$ respectively. Rear tire vertical stiffness $k_{vertical}$ and damping $c_{vertical}$ were set to 58570 N/m and 11650 $\mathrm{N \cdot s/m}$ respectively. They were also taken from [72] because they belong to a Suzuki GSX-T 1000 with the same type of tires as Aprilia Mana 850 GT. To calculate the rolling resistance force and as consequence the rolling resistance moment, presented in Fig. 5.9, it is used the equation 5.30.

$$F_{\text{rolling resistance}} = f_w \cdot F_z \tag{5.30}$$

Where f_w is the rolling resistance coefficient, set to 0.02 according to [68].

5.8 Road profiles

Different 2D road profiles were set with excel tables and connected with *Simcenter 3D Motion*, trying to replicate the ones used in testing applications in the *Aldenhoven* testing track. Figure 5.11, gives a top view of the referred proving ground and shows the created profiles on the software.

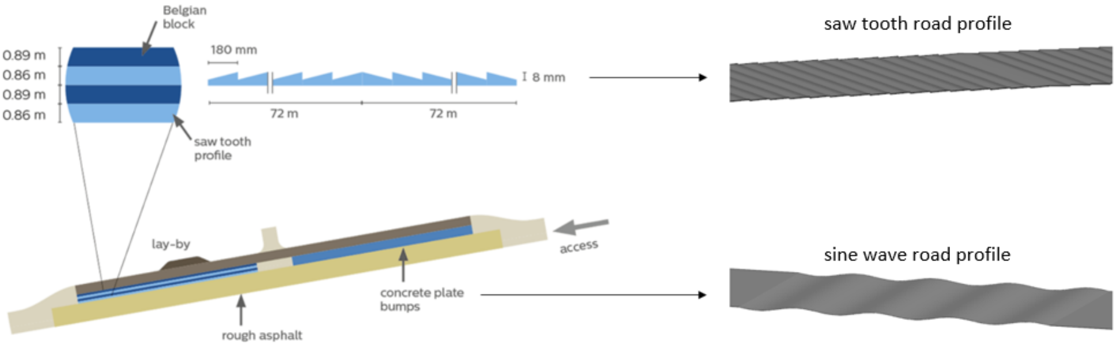


Figure 5.11: On the left: top view of *Aldenhoven* testing track; on the right: saw tooth road profile and sine wave road profile. Adapted from [73].

Chapter 6

Motorcycle braking system modeling

As previously mentioned in chapter 3, the work of this dissertation is centered around Aprilia Mana 850 GT motorcycle. The motorcycle braking system was the last part being modeled resorting the modeling tool *Simcenter Amesim* already introduced in section 2.5.2. A list of program's libraries were used for this construction process, namely the mechanical, signal and control, hydraulic (HYD) and hydraulic component design (HCD) libraries.

According to existing literature [36, 74], to the IGES files of Aprilia Mana 850 GT bought for visualization purposes and to the motorcycle catalogue [43] it was possible to conclude that the motorcycle's braking system is composed by rear and front master cylinders, ABS control, calipers, brake pads and disc brakes. Figure 6.1 schematize the Aprilia Mana 850 GT's braking system. Table 6.1 gives the available information on the braking system's calipers and disc brakes.

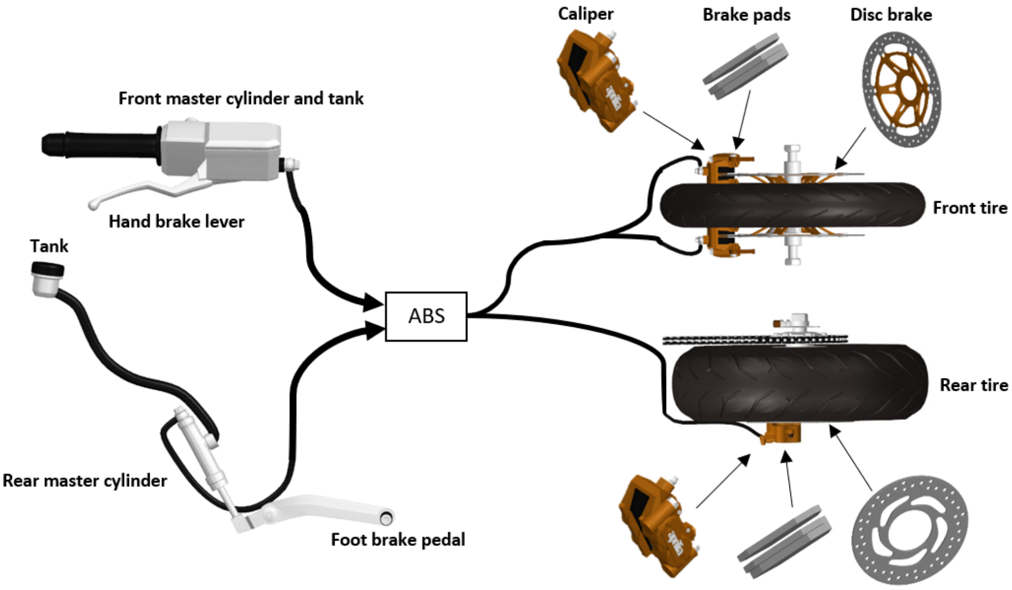


Figure 6.1: Aprilia Mana 850 GT's braking system scheme.

For the sake of simplification ABS control wasn't introduced in the braking system model.

Table 6.1: Braking system's calipers and disc brakes [45]

Brake wheel	**n° of disc brakes (diam.)**	**n° of calipers**	**n° of plungers per caliper (diam.)**
Front brake	2 (320 mm)	2	4 (2 with 27 mm, 2 with 32.03 mm)
Rear brake	1 (260 mm)	1	2 (25 mm)

6.1 Front and rear master cylinder

The first task is to relate the force applied by the driver on the hand brake lever and the foot brake pedal with the force originated in the front and rear master cylinder. In order to do that, two momentum equilibrium equations are set around the fulcrum points of both hand brake lever and foot brake pedal. Figure 6.2 show the forces and distances at stake.

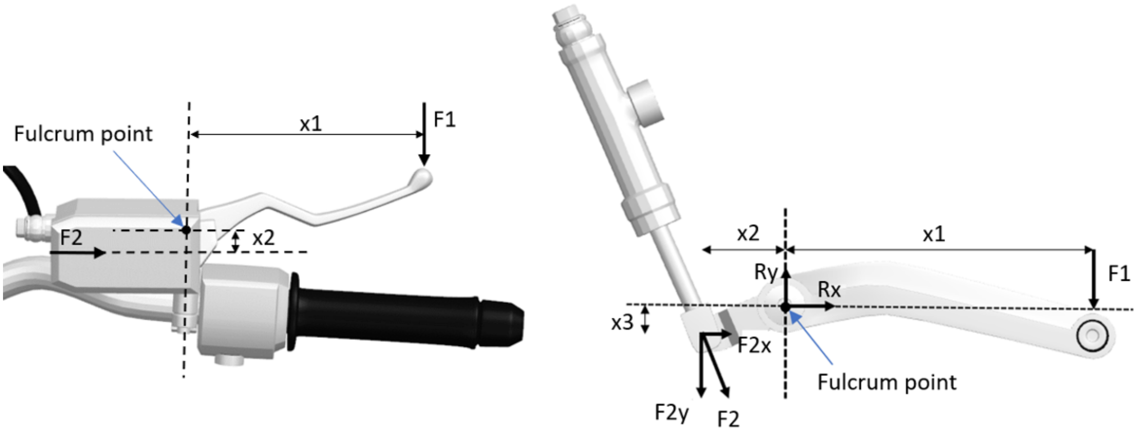


Figure 6.2: Forces applied on the hand brake lever (left) and foot brake pedal (right).

For $x1$ = 154 mm and $x2$ = 15 mm, the hand brake lever actuation $F1$ applies a force on the front master cylinder $F2$ of:

$$F1 \cdot x1 = F2 \cdot x2 \iff F2 = \frac{F1 \cdot 0.154}{0.015} \tag{6.1}$$

For $x1$ = 157 mm, $x2$ = 24.9 mm, $x3$ = 24.9 mm and the angle between $F2$ and $F2x$ being θ = 50°, the foot brake pedal actuation $F1$ applies a force on the rear master cylinder $F2$ of:

$$F1 \cdot x1 = F2 \cdot (cos(\theta) \cdot x3 + sin(\theta) \cdot x2) \iff F2 = \frac{F1 \cdot 0.157}{cos(\frac{50 \cdot \pi}{180}) \cdot 0.0249 + sin(\frac{50 \cdot \pi}{180}) \cdot 0.0249} \tag{6.2}$$

To model the front and rear master cylinders, as there were no information about the insides of these components, it was assumed they were constructed like presented in Fig. 6.3. They are composed by a spool, spring and an accumulator with hydraulic fluid. Figure 6.4 illustrates the *Simcenter Amesim* submodels used to replicate the front and rear master cylinder behavior.

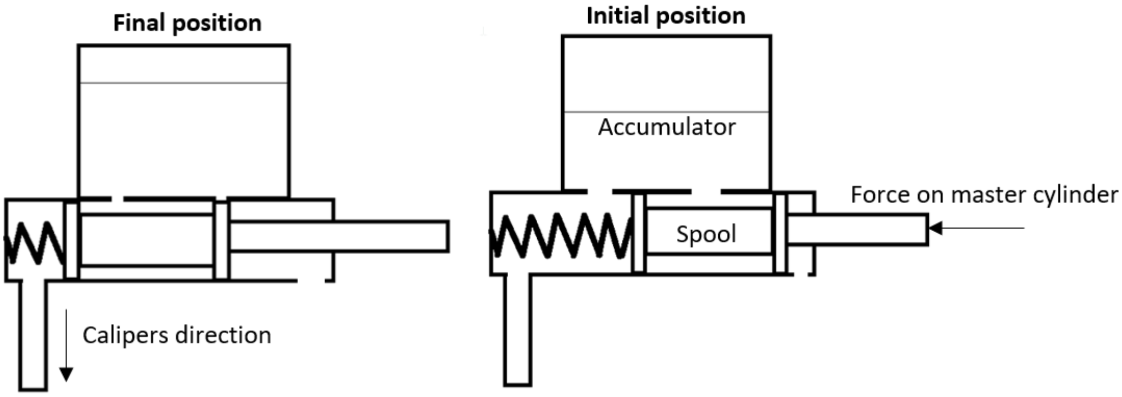


Figure 6.3: Scheme of front and rear master cylinder initial and final position after applied force.

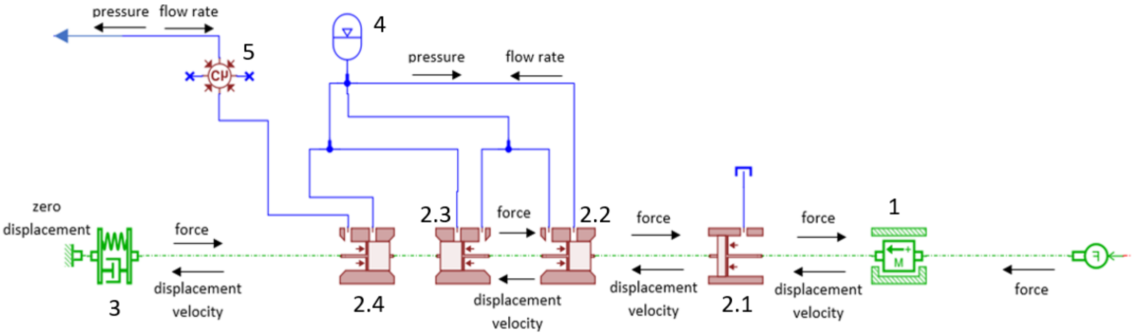


Figure 6.4: Master cylinder model.

Submodel 1 (see Fig. 6.4) represents the spool and its rod's mass. It relates the force acting on the master cylinder calculated with the help of equation 6.1 or equation 6.2, depending if its the rear or front cylinder, and the force from submodel 2.1 to compute the mass displacement and velocity, integrating equation 6.3.

$$acc = \frac{F_2 - F_1 + 9.81 \cdot mass \cdot sin(\theta)}{mass} \tag{6.3}$$

Where,

acc	= spool acceleration
θ	= spool inclination angle (set to 0°)
F_1	= force generated on the master cylinder
F_2	= force generated by submodel 2.1 on submodel 1
$mass$	= spool mass (set to 0.6 kg)

In order to represent the spool geometry it was divide in four submodels from 2.1 to 2.4 (see Fig. 6.4). Submodel 2.1 represents the right side of the spool and relates the force coming from submodel 2.2 and atmospheric pressure through the chamber hole with the force applied on submodel 1 as presented in equation 6.4.

$$F_1 = F_2 - p_{atm} \cdot \frac{\pi}{4}(dp^2 - dr^2) \tag{6.4}$$

Where,

F_1	= force applied on submodel 1
F_2	= force coming from submodel 2.2
dp	= spool diameter (set to 18 mm)
dr	= spool rod diameter (set to 2 mm)

It is also possible to compute the instant volume of the chamber resorting the displacement coming from submodel 1 and chamber lenght at zero displacement and compute the instant flow rate at the hole resorting the velocity coming from submodel 1. Submodels 2.2, 2.3 and 2.4 are similar to 2.1. However the combination of these 3 submodels allow also the definition of the chambers' holes diameters and the initial and final position of the spool, i.e. which holes are closed or opened to submodel 4, the accumulator, throughout the simulation. The internal spool rod diameter was set to 5 mm (this value was applied to submodel 2.2 and 2.3) and because there is no rod in between the spring, the spool rod diameter of submodel 2.4 was set to 0 mm. Submodel 3 represents the spring stiffness and the systems' damping whereas submodel 5 represents the dead volume inside the chamber of submodel 2.4. Besides this, submodel 5 transforms the input flow rate into the final output braking fluid pressure sent to the calipers.

6.2 Calipers

Depending on the fluid pressure that comes from the master cylinder, the plungers[1] inside the calipers apply a bigger or smaller normal force on the brake pads, which cause them to move and become in contact with the disc brake, creating a resistive torque on the wheel. Figure 6.5 illustrates the *Simcenter Amesim* submodels used to replicate a caliper's plunger behavior. As explained before in section 6.1 submodel 1 is replicating the dead volume of submodel 3's chamber.

[1] hydraulic cylinder without piston, just rod.

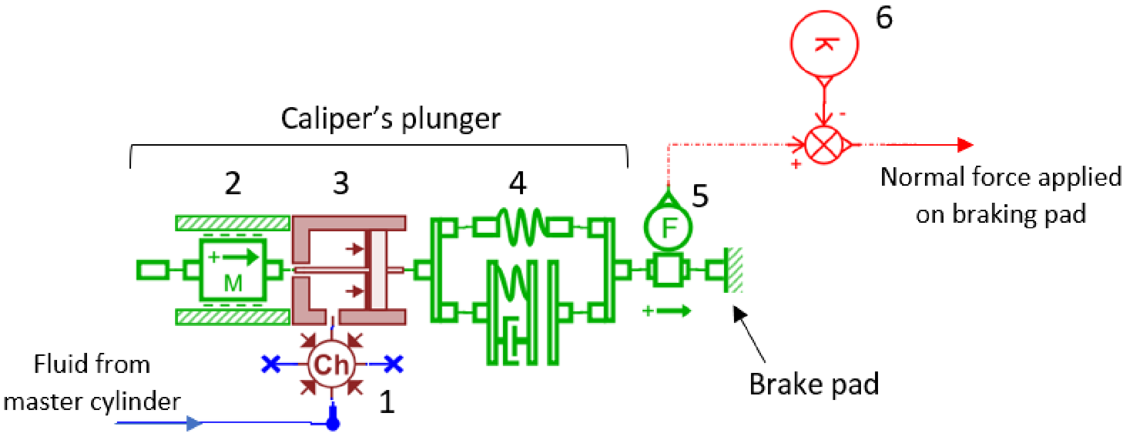

6
Caliper's plunger
2
3
4
5
M
F
Ch
1
Normal force applied on braking pad
Fluid from master cylinder
Brake pad

Figure 6.5: Caliper's pungler model.

Also discussed in the previous section, submodel 2 represents the plunger's mass and submodel 3 the plunger's chamber and rod (piston rod diameter is set to 0 mm leaving just the piston diameter to define the plunger). Submodel 4 is composed by two submodels [19]: one spring with the plunger mechanical equivalent stiffness mainly due to the hydraulic stiffness of the brake hoses; an elastic contact model to simulate elastic effects during contact between plunger and brake pad. Submodel 5 is a force sensor meant to extract the normal force applied by the plunger on a brake pad. Submodel 6 is an offset to remove the initial force provoked by the initial fluid pressure sent by the master cylinder (when no force is applied by the rider). This offset needs to be inserted because while in this caliper's plunger model the brake pad and plunger are always in contact, in reality, the two components are not in contact with each other when no force is applied by the driver and thus the normal force output in the brake pad should be null. Based on Tab. 6.1 information for the whole front braking system model, 4 caliper's plunger models (Fig. 6.5) were used: two for each caliper where one plunger has a piston diameter of 32.03 mm and other with 27 mm. The output normal force applied on the pads by these two plungers is multiplied by 2 to complete the total influence of the 4 plungers per caliper.

As far as the rear braking system is concerned, the model has 1 caliper's plunger model with a piston diameter of 25 mm and the output normal force applied on the pad by this plunger is multiplied by 2 to complete the couple of caliper's plungers total influence.

6.3 Friction model and final model overview

To simulate friction between the brake pads and the disc brakes when in contact with each other, and obtain the braking torque on the front and rear wheel of the motorcycle, the LuGre friction model was used. The LuGre model is capable of simulating the memory dependent behavior of friction and phenomenons such as predisplacement, rate-dependence and hysteresis which have been experimentally identified. So, basically, while static friction models are appropriate when we have steady-state conditions for the linear and angular velocities, the LuGre model capture the transient friction behavior under time-varying velocity conditions being a so-called "dynamic friction model" (just like the Dahl model) [75]. However, it can be said that the LuGre model is an extension of the Dahl model because can also capture the Stribeck effect[2] and thus can describe stick-slip motion[3] [76]. In this model, friction is defined as the force necessary to bend bristles, behaving like springs, as presented in Fig. 6.6. The internal state of the model is the average bristle deflection z. The LuGre model is described by equations 6.5 and 6.6.

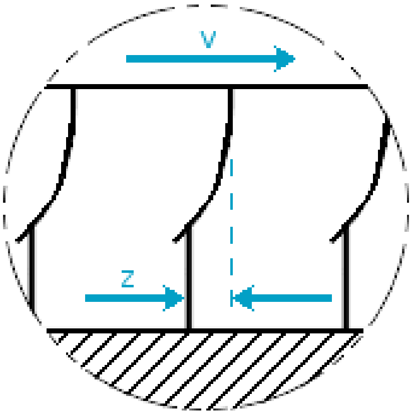


Figure 6.6: Microscopic view of bristle deflection [19].

$$\dot{z} = v - \sigma_0 \cdot \frac{|v|}{g(v)} \cdot z \tag{6.5}$$

$$F = \sigma_0 \cdot z + \sigma_1 \cdot \dot{z} + f(v) \tag{6.6}$$

Where,

v	= velocity between two surfaces
F	= friction force
σ_0	= stiffness of the bristles
σ_1	= bristle microscopic damping coefficient
$f(v)$	= viscous friction (macrodamping)

In equation 6.6, viscous friction $f(v)$ can be calculated multiplying viscous friction coefficient σ_2 and velocity v. In equation 6.5, $g(v)$ captures Coulomb friction and the Stribeck effect. A reasonable choice of $g(v)$ giving a good approximation of the Stribeck effect is the following:

[2] Phenomenon in which for low velocities between surfaces the friction force decreases with increasing velocity.

[3] Stick-slip motion manifests itself as repeated sequences of sticking between two surfaces with static friction followed by sliding or slipping of the two surfaces, for instance, when moving slowly, machines are likely to exhibit stick-slip motion [19].

$$g(v) = F_c + (F_s - F_c) \cdot e^{-|v/v_s|^{\alpha}} \tag{6.7}$$

Where,

F_c = Coulomb friction force

F_s = Stiction force

v_s = Stribeck constant

α = Stribeck constant

Stiction and Coulomb friction forces can be calculated multiplying stiction friction coefficient σ_3 and Coulomb friction coefficient σ_4 with normal force applied by one surface on another, respectively. LuGre friction model can be characterized by the steady-state friction function represented in Fig. 6.7.

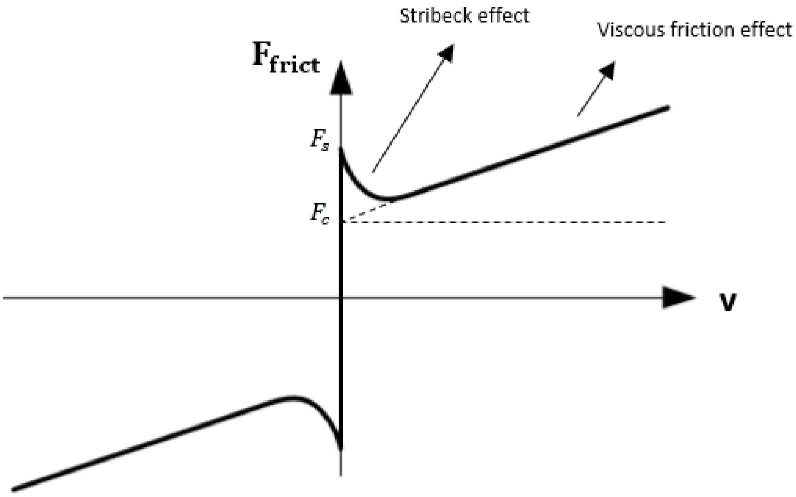


Figure 6.7: Relation between LuGre model friction force and relative velocity between surfaces [77].

Parameters referred in the previous equations throughout this section can only be find by means of experimental results. As there is no available experimental results for LuGre friction model between these specific motorcycle's brake pads and disc brakes, it was used the parameters available in Tab. 6.2.

Table 6.2: LuGre friction model parameters [78, 19].

σ_0 [Nm/rad]	σ_1	σ_2 [Nm/(rad/s)]	σ_3	σ_4	α	v_s [rev/min]
1000	0	0.13	0.45	0.4	2	20

Figure 6.8 and Fig. 6.9 show the final front and rear braking system models respectively.

The submodel used to implement the Lugre model needs speed feedback from the front and rear wheel and the diameter on which friction acts (contact between brake pad and disc brake) to compute the front and rear braking torque. The diameter on which friction acts, for the front wheel, was set to 316 mm and for the rear wheel was set to 256 mm.

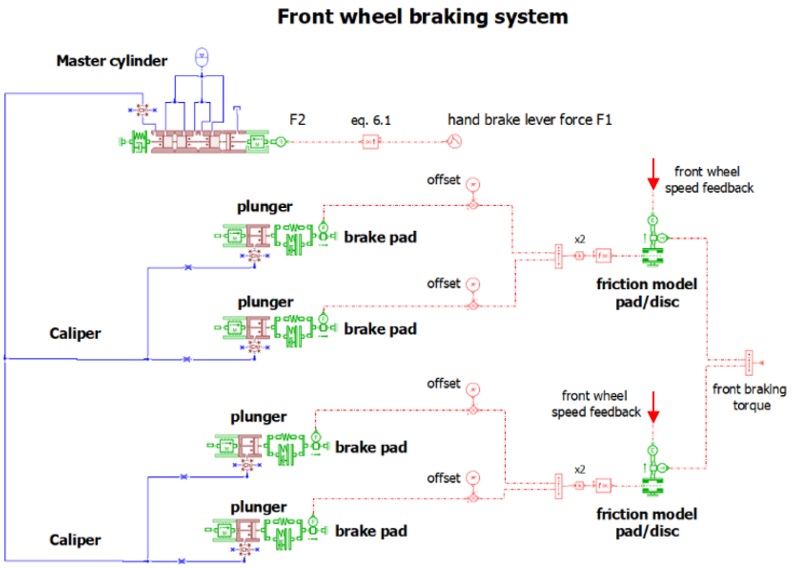


Figure 6.8: Front braking system model.

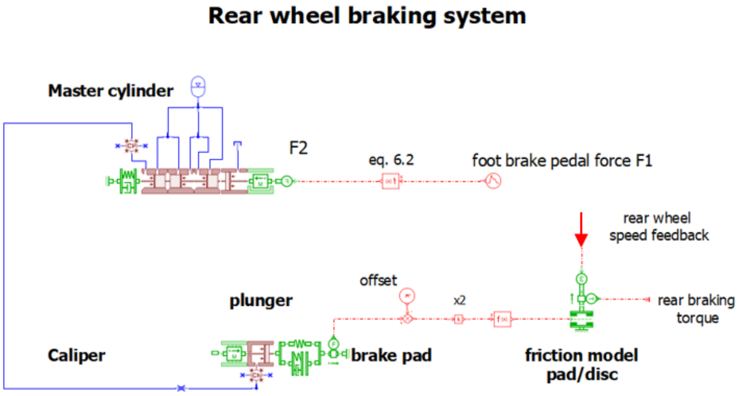


Figure 6.9: Rear braking system model.

In order to get a reasonable front and rear wheel speed feedback the braking system models shown in Fig. 6.8 and Fig. 6.9 were connected with the model constructed in chapter 4 and presented in Fig. 4.22. A connection between the motorcycle model, shown in figure 4.22, and the front braking system model, is made and presented in Fig. 6.10. In this connection was also added a switch to replicate the contact between pad/disc - output front braking torque is null if no hand brake lever force is applied.

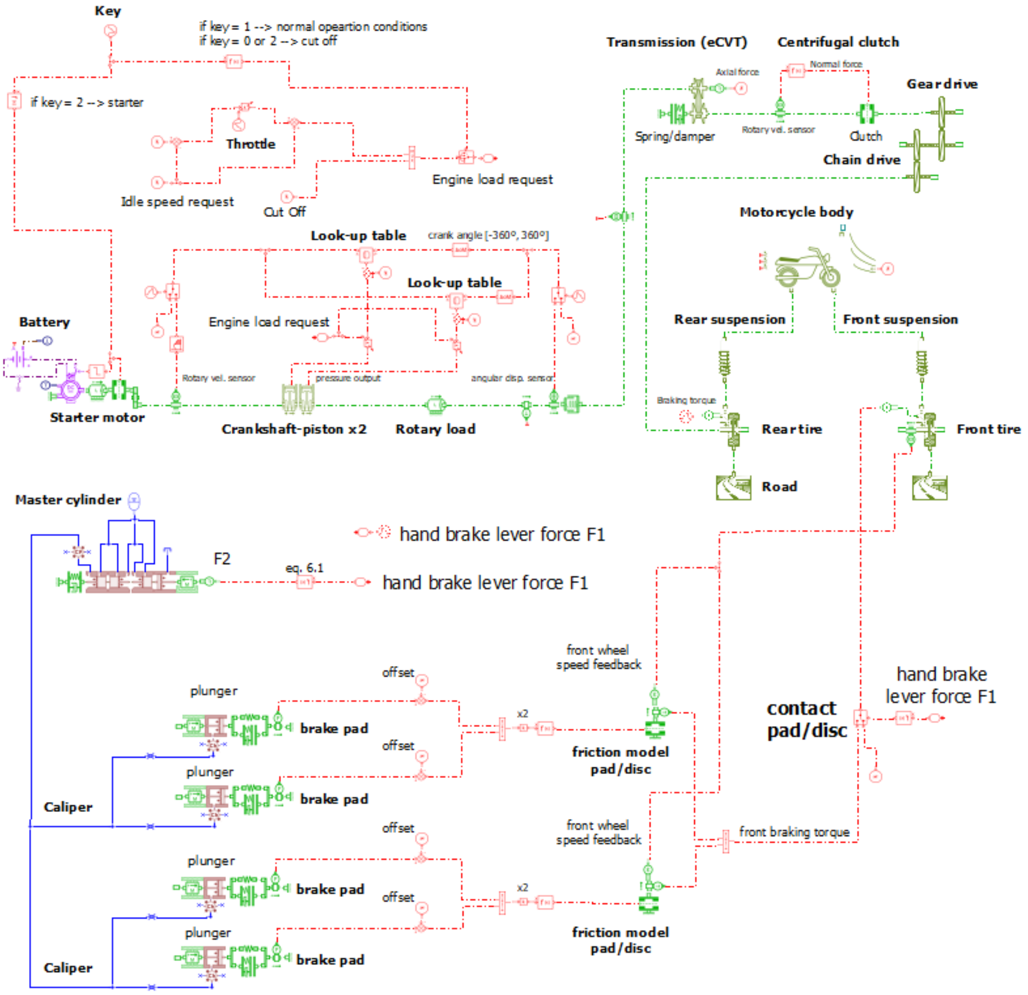


Figure 6.10: Connection between model from figure 4.22 and front braking system model presented in figure 6.8

Chapter 7

Results

Throughout this chapter results from 3 created models (engine, multibody and braking system model) were presented and discussed afterwards. The stated models are characterized as behavioral models. As a consequence, the main goal is to conclude if the models' results represent at some degree an already observed behavior in existing literature.

7.1 Engine model

The main purpose of this model, represented in Fig. 3.7, is computing the output engine torque, based on an input throttle command, to be sent in real-time to the eCVT bench and receiving as input a speed feedback from the transmission itself.

Before the engine model is suited for real-time, it was needed to model the rest of the engine parts (explained throughout section 4.2) to check if the engine's model itself was working correctly.

A 45 seconds simulation was made to observe the engine and overall motorcycle's parts behavior. Figure 7.1 relates key, throttle, and braking torque inputs with the engine speed and the motorcycle center of gravity (COG) velocity. The simulation starts with the motorcycle on a standing position with the engine off and the key at position 0. After 0.8 seconds the engine is turned on by turning the key into position number 2 during 2 seconds. Consequently the engine's starter increases the engine speed until it reaches 1400 rpm idle speed. Following this event, key signal goes to position 1 which means from now on the engine is in its normal operation phase. The engine is at idle speed until second 9 of simulation where a wide throttle input signal (signal equal 1) is set during 12 seconds. Because of this, the engine speed increases until 8500 rpm as well as the motorcycle COG speed which reaches 140 km/h. According to Fig. 7.1 the motorcycle goes from 0 to 100 km/h approximately in 8 seconds, which seems to be reasonable comparing with acceleration charts in [79], where it goes from 0 to 100 km/h in 5.3 seconds. It is important to point out that the acceleration charts in [79] were taken with an optimized eCVT ratio control (this

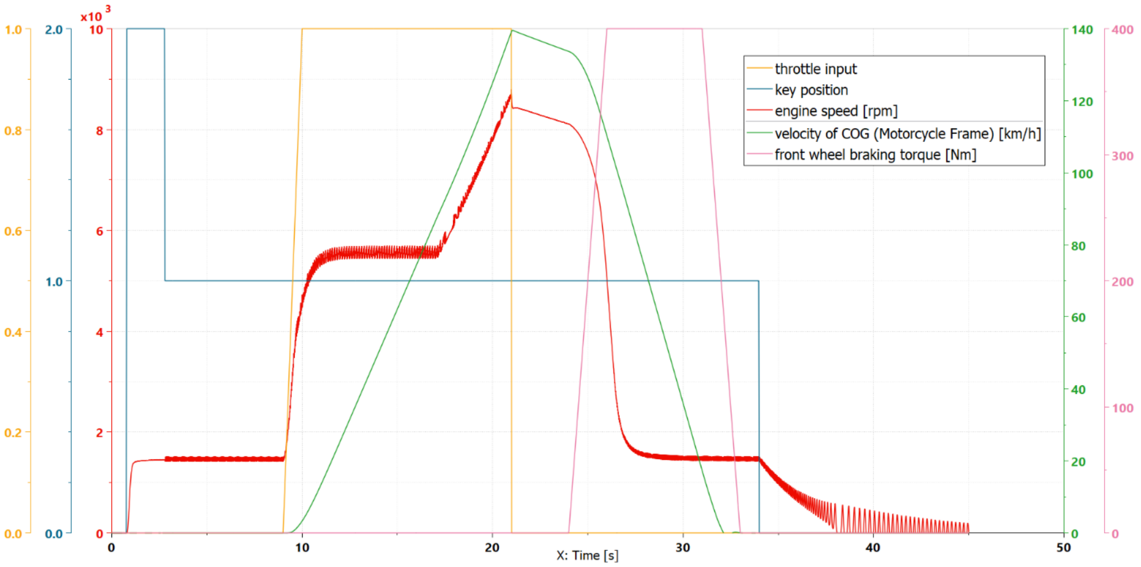


Figure 7.1: Engine speed and velocity of motorcycle COG based on throttle input, key position and braking torque.

ratio can go from 0.831 to 2.079 [45]) while this simulation was made with a constant transmission ratio of 1.2685 (see section 4.2.5). Following the end of the throttle input signal, the motorcycle slowly decreases its speed while no input braking torque is applied. Next, a 400 Nm and a 200 Nm braking torque is set to the front and rear wheel respectively, hence the motorcycle speed decreases until stopping completely while the engine reaches idle speed. At second 34 of simulation the engine is turned off by setting the key position to 0 and therefore the engine speed eventually reaches 0 rpm.

From Fig. 7.1 is also possible to observe the non-linear increase of the engine speed while applying a wide open throttle input. This behavior is typical for CVT (continuously variable transmission) motorcycles and can be explained in the following steps [80]:

1. After opening the throttle while idling the engine speed rises and the centrifugal clutch starts to transmit torque. This can be called **contact point**.

2. Posterior to this, centrifugal clutch is slipping while transmitting the given engine torque.

3. The centrifugal clutch's drum speed starts increasing until it matches the eCVT driven shaft speed. Until this point in time the engine is running at constant speed called **slipping speed** and the vehicle is accelerating.

4. When the centrifugal clutch's drum speed matches the eCVT driven shaft speed the friction between the rotational bodies of the clutch changes from sliding to static friction. No differential speed occurs in the clutch and therefore this point is called **lockup point**.

5. From this point on the motorcycle speed increases proportionally to the engine speed.

Figure 7.2, shows the speed of centrifugal clutch's drum and eCVT driven shaft speed throughout the first 21 seconds of the simulation presented in Fig. 7.1 that proves the steps explained above [81].

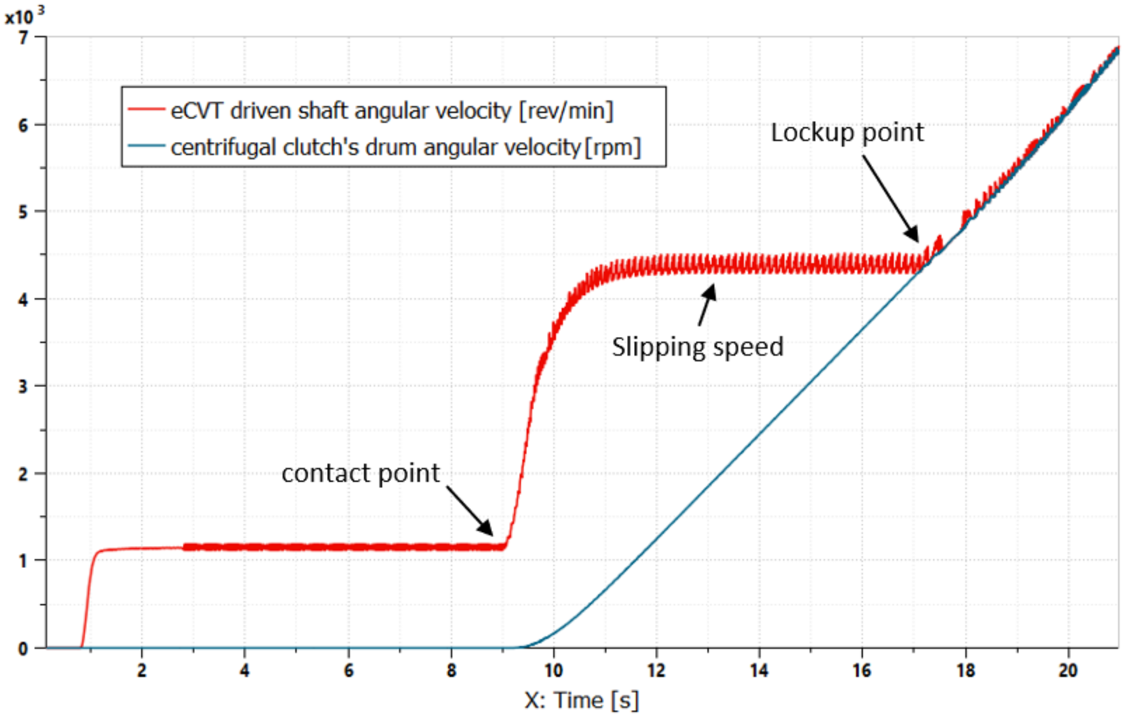


Figure 7.2: Motorcycle's eCVT typical behavior during firs 21 seconds of simulation (acceleration).

In the real motorcycle exists a transmission ratio control that during an acceleration of this type (wide open throttle acceleration) creates a behavior similar to the one presented in Fig. 7.3. The behavior of the engine speed curve presented in Fig. 7.1 is similar to the one presented in Fig. 7.3 from point 1 to point 4. The difference is that for the real motorcycle the initial eCVT transmission ratio is the lowest gear, i.e. the max ratio 2.079 and not the constant ratio set in the simulation equal to 1.2685. From point 4 to point 5 the transmission control changes the ratio from the lower gear (2.079) to the higher gear (0.831) keeping the engine speed at a constant spot for optimal performance.

Now that the general behavior of the motorcycle seems to be reasonable, the engine output torque can be seen in Fig. 7.4. It shows a characteristic form similar to the instantaneous output torque of a V-twin engine [42].

7.2 Multibody model

The main purpose of this model, represented in Fig. 3.7, is computing the the wheels' angular velocity to be sent to the braking system model and the eCVT driven shaft's angular velocity to

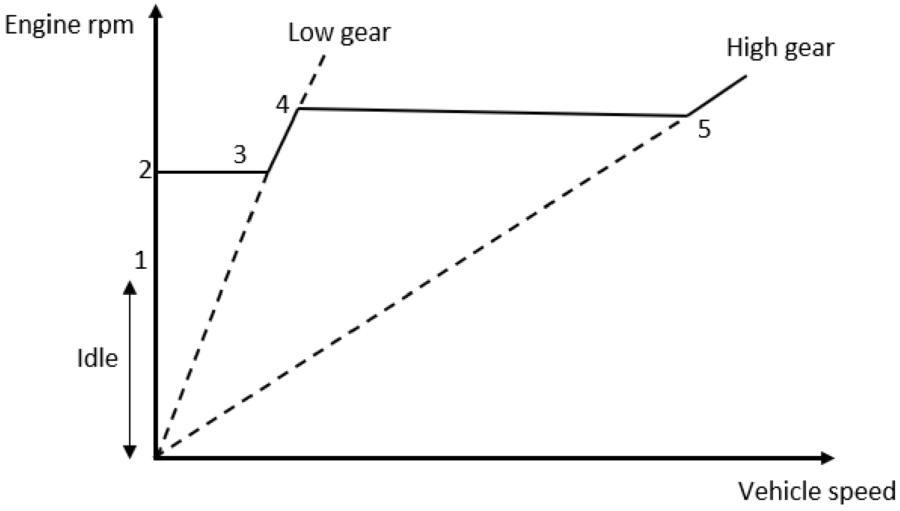


Figure 7.3: Real motorcycle's eCVT typical behavior with transmission ratio control.

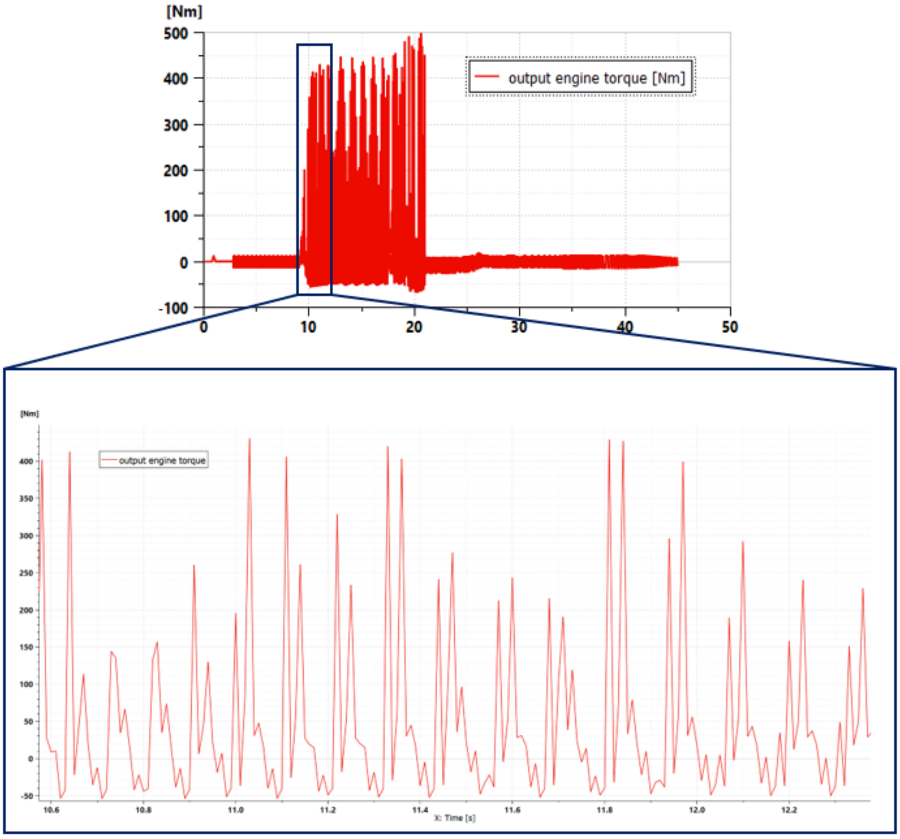


Figure 7.4: Output engine torque during the 45 seconds simulation.

be sent to the real eCVT, based on an input torque on the eCVT driven shaft coming from the real eCVT.

Before the multibody model is suited for real-time it was necessary to check its behavior. In order to do this, a constant input torque was applied on the driven eCVT shaft (dummy shaft, see chapter 5) and several variables throughout a 10 second simulation were studied for sine wave road profile (Fig. 7.5):

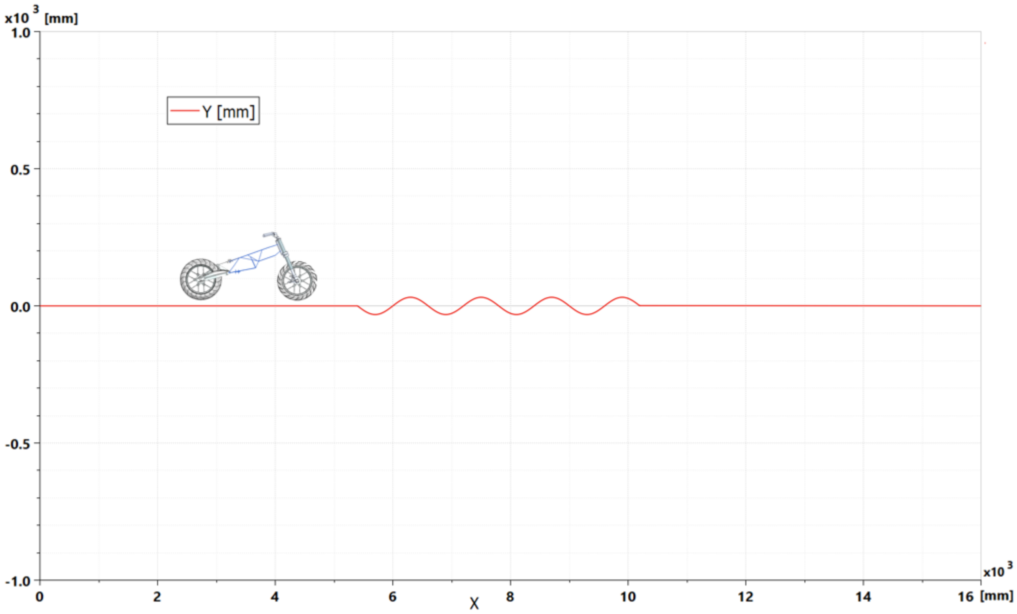


Figure 7.5: Sine wave road profile (XZ plane).

- Rear frame pitch angle, vertical displacement and longitudinal velocity (Fig. 7.6, Fig. 7.7 and Fig. 7.8 respectively);
- Front and rear suspension relative displacement (Fig. 7.9);
- Front tire longitudinal and normal forces (Fig. 7.10 and Fig. 7.11 respectively) ;
- Rear tire longitudinal and normal forces (Fig. 7.10 and Fig. 7.11 respectively).

From the previous graphs, the following can be concluded:

1. A initial decrease in pitch angle and vertical displacement of the rear frame (Fig. 7.6 and Fig. 7.7) and front and rear suspension relative displacement (Fig. 7.9) comes from the fact that the motorcycle initial position is not at road level, so in the initial period of simulation the motorcycle is falling because of gravity and the decrease of these variables comes when the motorcycle and road start being in contact. After this, there is a rebound behavior thanks to the motorcycle suspension.
2. At second 1 of simulation the motorcycle reaches the first bump and passes all of them in 1 second. It is visible in all curves the spikes provoked by the bumps.

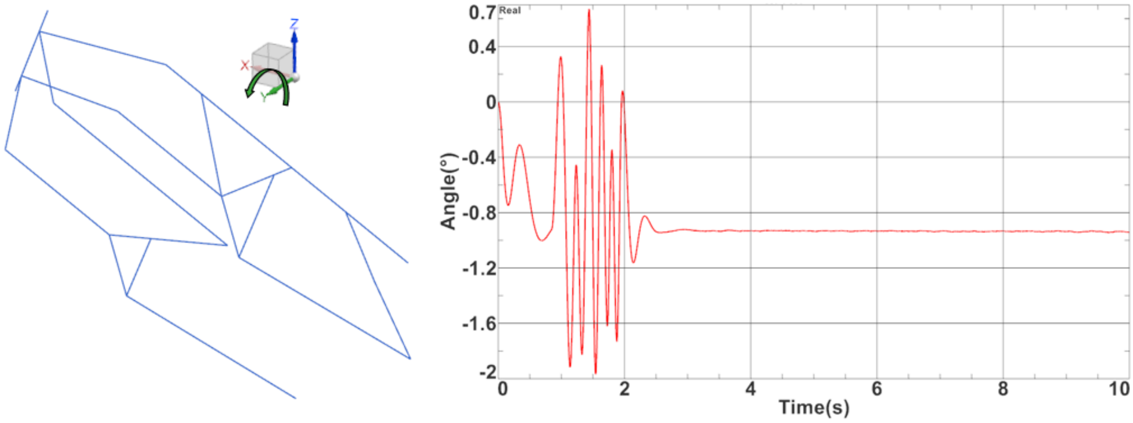


Figure 7.6: Rear frame pitch angle.

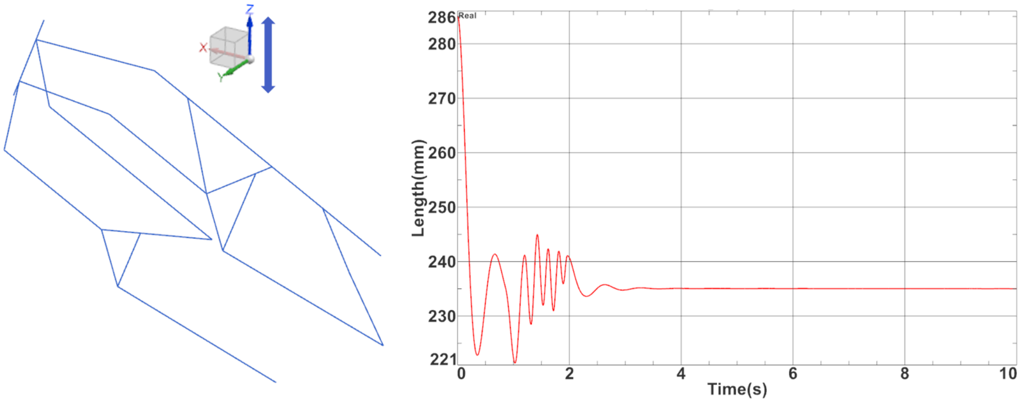


Figure 7.7: Rear frame vertical displacement.

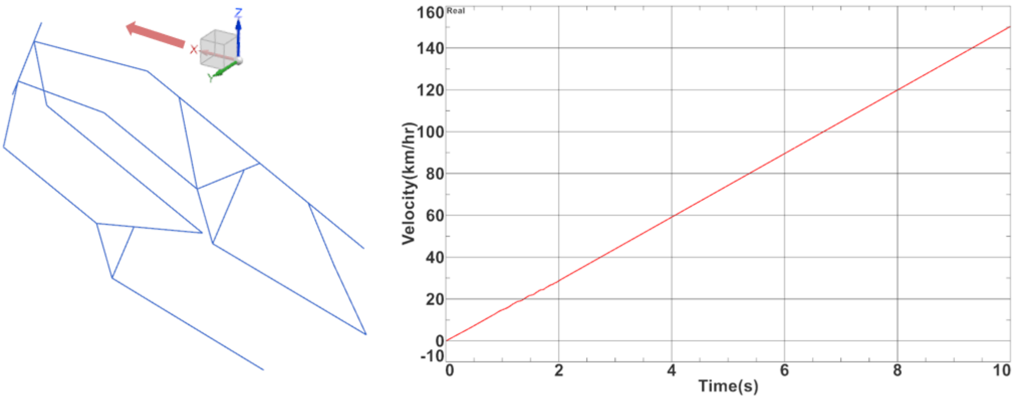


Figure 7.8: Rear frame longitudinal velocity.

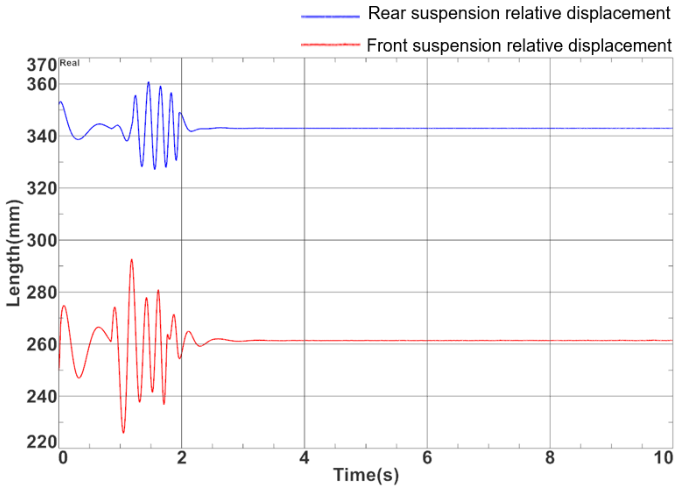


Figure 7.9: Front and rear suspension relative displacement.

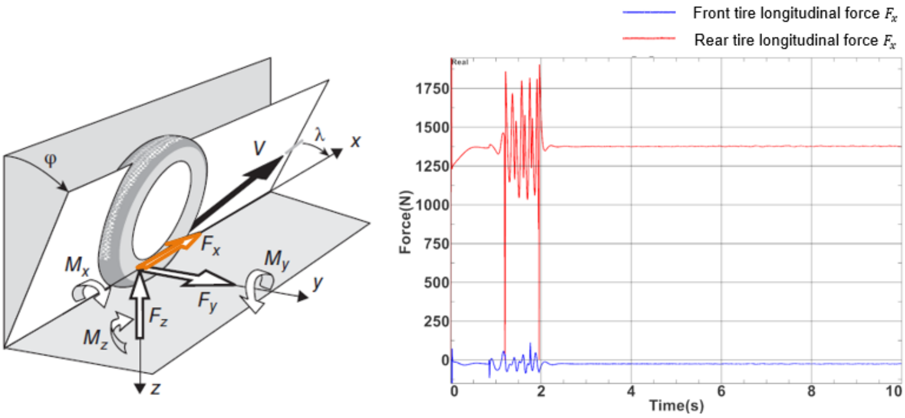


Figure 7.10: Front and rear tire longitudinal force.

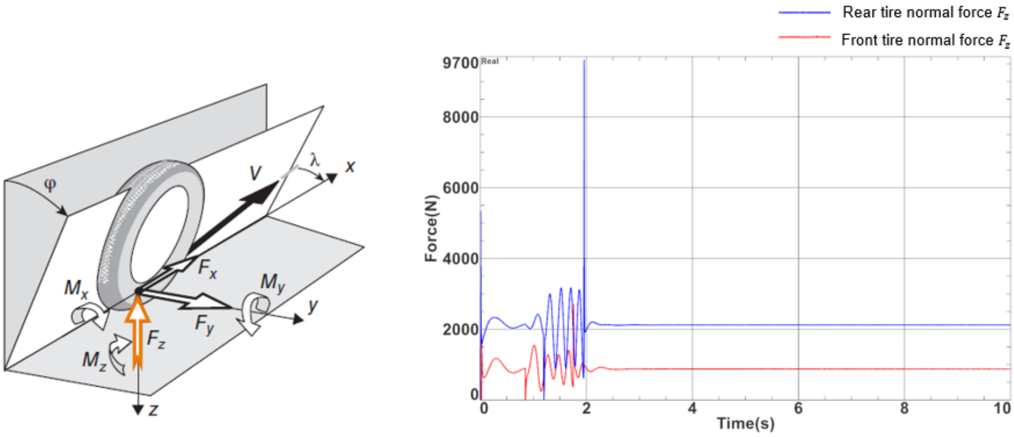


Figure 7.11: Front and rear tire normal force.

3. After the bumps, motorcycle reaches an equilibrium position and remains like that until the end of simulation, as the road keeps a flat shape.

4. The rear tire longitudinal force in equilibrium state has a positive value of 1375 N while the front tire longitudinal force has a negative value of -25 N which replicates the behavior seen in [72].

5. The rear and front tires normal force are 2120 N and 875 N respectively, in equilibrium position.

7.3 Braking system model

The main purpose of this model, represented in Fig. 3.7, is computing the front and rear wheel braking torque to be sent to the motorcycle multibody model and receive from this same model the front and rear wheel speed feedback.

Before this model is suited for real-time, it was necessary to check its behavior first, so the front braking system model was connected with the model presented in Fig. 4.22 giving origin to the model shown in Fig. 6.10 as also explained in section 6.3.

A 50 second simulation was made including the front braking system model. A hand brake lever force of 50 N was applied between second 32 and 43 resulting in a corresponding front braking torque. Figure 7.12 relates the front braking torque with the input hand brake lever force and the front wheel rotary velocity feedback. From Fig. 7.12 it is possible to notice that even after the front wheel is completely stopped (around second 42 of simulation) still exists an applied braking torque. This behavior can be explained with the motorcycle pitch angle (represented in

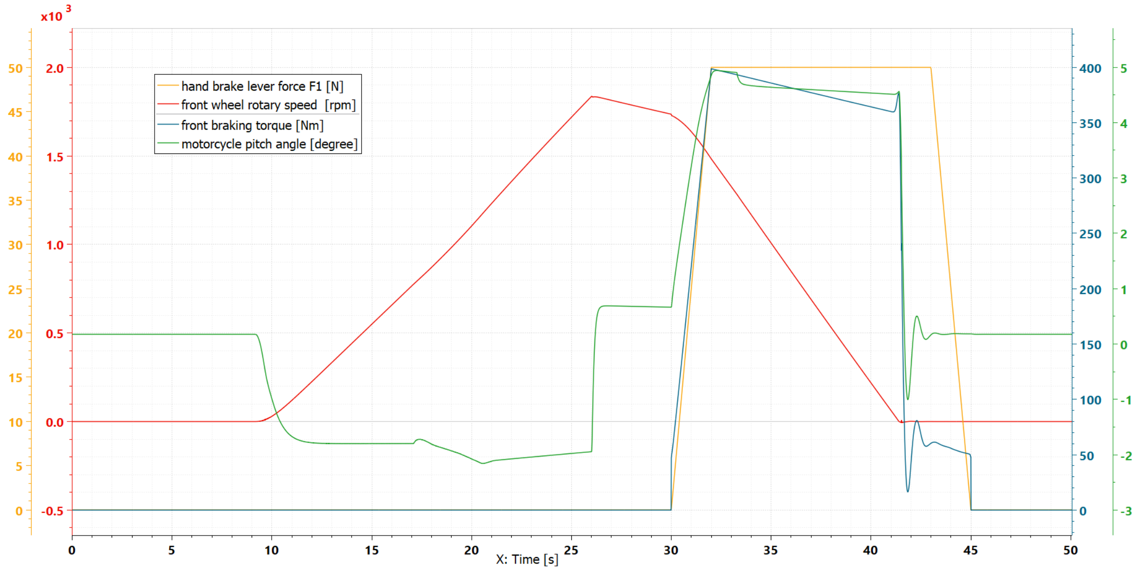


Figure 7.12: Front braking torque based on the input hand brake lever force and the front wheel rotary velocity feedback.

green in Fig. 7.12): During deceleration (when braking torque is applied), the load on the front wheel increases, while that on the rear wheel decreases and thus there is a load transfer from the rear to the front wheel, increasing the pitch angle to 5 degrees [68]. When the motorcycle wheels are considered completely stopped (wheel rotary velocity equals to 0 rpm) the motorcycle body stills exhibits an oscillating behavior before reaching equilibrium position and thus creating a residual braking force.

Figure 7.13 shows the plot of friction torque relative to the front wheel rotary velocity. In fase

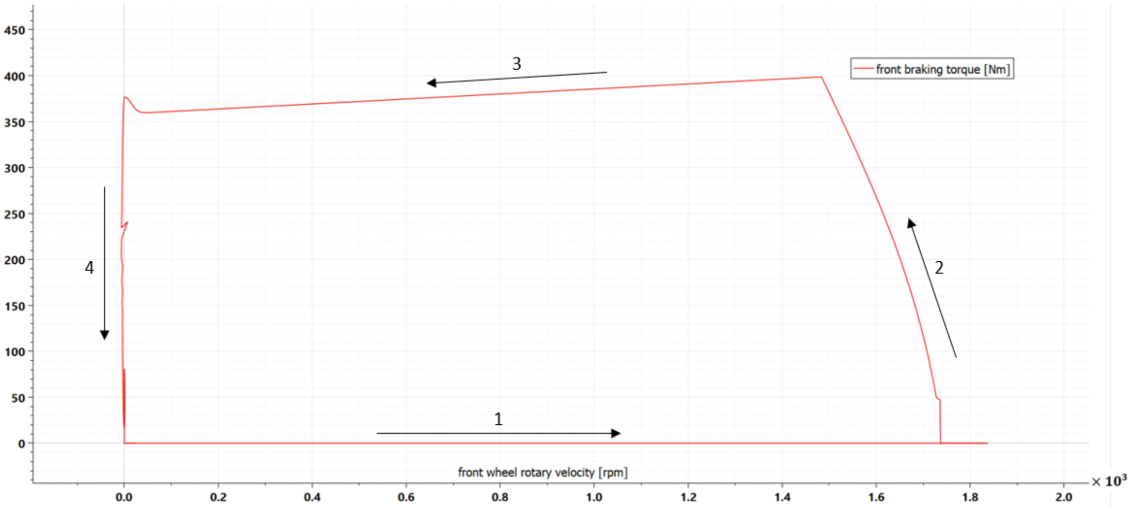


Figure 7.13: Front braking torque as function of front wheel rotary velocity feedback.

1 motorcycle is increasing its velocity and no braking torque is applied. Fase 2 is characterized by the abrupt increase of braking torque as a consequence of the force applied by the driver on the hand lever. Fase 3 and 4 shows the typical behavior of a LuGre friction model analyzed in section 6.3 and shown in Fig. 6.7, where Stribeck effect, viscous friction effect, stiction friction and Coulomb friction can all be observed.

Chapter 8

Conclusions and Future Work

Hybrid testing and the model based development approach, in the four and two wheeler vehicle industry, have been increasingly used in the early stages of the development phase and design. Following this line of research, by mixing numerical simulations with real physical components, this dissertation aimed for the development of innovative hybrid testing solutions, resorting to high modeling software tools, namely *Simcenter Amesim* and *Simcenter 3D Motion*. To accomplish this goal prior work had to be done - an Aprilia Mana 850 GT motorcycle transmission, more specifically an eCVT (electronically controlled continuous variable transmission), was transformed into a test bench. Furthermore, to combine the physical component (transmission) with numerical simulations and fully reach an eCVT-in-the-loop stage, a real-time platform was employed.

During this dissertation it were developed the following motorcycle subsystems models:

- An Engine model resorting to *Simcenter Amesim*, to replicate engine torque and rotary velocity based on a throttle input, whose creation was divided in two different steps - firstly, an auxiliary engine model was built, so that look-up tables could be produced; then, centered around the newly look-up tables, the final engine model construction took part. Overall, a set of submodels were included, simulating a big group of engine characteristics, as pressure dynamics, engine geometry, cylinders combustion process, among many others, allowing features like idle speed and engine starter to be acountable. A complete *Simcenter Amesim* motorcycle model was required to give proper engine load and speed.

- A Multibody model resorting to *Simcenter 3D Motion*, to replicate motorcycle dynamics and contact between the tires and the road. To acomplish these features, Pacejka's "Magic Formula" was implemented for tire modeling, 3D modelling and assembling of motorcycle bodies were executed, as well as different road profiles based on the Aldenhoven testing track.

- A Braking System model, to replicate front and rear motorcycle braking torques, through the use of a dynamic friction model (LuGre) between the brake pads and disc brakes.

All models are charaterized as behavioral, due to the lack of experimental data around the Aprilia Mana 850 GT, and, as a consequence, it was impossible to analyze the degree of accuracy shown by simulations results. Instead, different conclusions were taken, comparing obtained simulation results with already observed motorcycle behaviour in existing literature:

- Typical CVT behaviour was observed during a 45 seconds simulation of the complete *Simcenter Amesim* motorcycle model, showing centrifugal clutch slipping phenomenon and similar V-twin engine instantaneous output torque.
- A 10 seconds multibody simulation was performed to understand how suspensions, rear frame and tires behave when the motorcycle goes through a sine wave road profile.
- A 50 second simulation of the Engine model and Braking System model combined, demonstrated the characteristic function of a LuGre friction model.

8.1 Future Work

In order to keep the project moving forward and reach the final goal, which is to perform eCVT-in-the-loop real-time hybrid testing simulations, the following tasks need to be performed:

1. When the complete physical motorcycle arrives there will be a possibility to increase all models' accuracy with increased experimental data and measurements of different motorcycle parts (for example the engine valvetrain).
2. Making the models suitable for real-time. For now, models are running with a standard integrator type and the goal is to make them run with fixed-time step integrator. Model identification and simplification, addressed in section 2.4.5, can be needed to get reasonable fixed-time steps.
3. Creation of FMU (functional mock-up unit) for every designed model, so that all system units can communicate with each other during co-simulations, even if they were developed in different software packages.
4. Upload models into the real-time platform, explained in section 3.4, and compare the results with the offline ones observed in the software packages. Results should be compatible.
5. The upload and connection between models and the connection between models and the physical eCVT test bench is expected to be a gradual process and future setbacks and model modifications can happen.

References

[1] P. Benson Shing, *Real-Time Hybrid Testing Techniques*. Vienna: Springer Vienna, 2008, pp. 259–292. [Online]. Available: https://doi.org/10.1007/978-3-211-09445-7_6

[2] E. Bringmann and A. Kramer, *Model-Based Testing of Automotive Systems*, 2008.

[3] D. Bhatt, B. Hall, S. Dajani-Brown, S. Hickman, and M. Paulitsch, "Model-based development and the implications to design assurance and certification," in *24th Digital Avionics Systems Conference*, vol. 2, Conference Proceedings, p. 13 pp. Vol. 2.

[4] M. Törngren and O. Larses, "Characterization of model based development of embedded control systems from a mechatronic perspective; drivers, processes, technology and their maturity," 2004.

[5] "Model-based system testing: Efficiently combining test and simulation for model-based development," White paper, Siemens Digital Industries Software, 2019.

[6] M. Sarrazin, H. Van der Auweraer, and F. Marques dos Santos, *Model Based System Testing: A new drive to integrating test and simulation*, 2017.

[7] F. Marques dos Santos, R. Pastorino, B. Peeters, C. Faria, W. Desmet, L. Goes, and H. Van der Auweraer, *Model Based System Testing: Bringing Testing and Simulation Close Together*, 2016.

[8] A. Albers, M. Behrendt, J. Schroeter, S. Ott, and S. Klingler, *X-in-the-Loop: A Framework for Supporting Central Engineering Activities and Contracting Complexity in Product Engineering Processes*, 2013.

[9] H. Gao, T. Zhang, H. Chen, Z. Zhao, and K. Song, "Application of the x-in-the-loop testing method in the fcv hybrid degree test," *Energies*, vol. 11, p. 433, 2018.

[10] G. Tibba, C. Malz, C. Stoermer, N. Nagarajan, L. Zhang, and S. Chakraborty, *Testing automotive embedded systems under X-in-the-loop setups*, 2016.

[11] V. Ivanov, K. Augsburg, C. Bernad, M. Dhaens, M. Dutré, S. Gramstat, P. Magnin, V. Schreiber, U. Skrt, and N. Van Kelecom, "Connected and shared x-in-the-loop technologies for electric vehicle design," *World Electric Vehicle Journal*, 2019.

[12] P. A. Laplante and S. J. Ovaska, *Real-time systems design and analysis: tools for the practitioner*. John Wiley and Sons, 2011.

[13] V. Schreiber, V. Ivanov, K. Augsburg, M. Noack, B. Shyrokau, C. Sandu, and P. S. Els, "Shared and distributed x-in-the-loop tests for automotive systems: Feasibility study," *IEEE Access*, vol. 6, pp. 4017–4026, 2018.

[14] J. Bélanger, P. Venne, and J.-N. Paquin, "The what, where, and why of real-time simulation," *Planet RT*, pp. 37–49, 2010.

[15] G. Stettinger, J. Zehetner, M. Benedikt, and N. Thek, "Extending co-simulation to the real-time domain," in *SAE 2013 World Congress and Exhibition*, 2013, Conference Paper. [Online]. Available: https://doi.org/10.4271/2013-01-0421

[16] G. Schweiger, C. Gomes, G. Engel, I. Hafner, J. Schoeggl, A. Posch, and T. Nouidui, "An empirical survey on co-simulation: Promising standards, challenges and research needs," *Simulation Modelling Practice and Theory*, vol. 95, pp. 148–163, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1569190X1930053X

[17] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: A survey," vol. 51, no. ACM Comput. Surv., p. Article 49, 2018. [Online]. Available: https://doi.org/10.1145/3179993

[18] A. Ben Khaled, M. Ben Gaid, N. Pernet, and D. Simon, "Fast multi-core co-simulation of cyber-physical systems: Application to internal combustion engines," *Simulation Modelling Practice and Theory*, vol. 47, pp. 79–91, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1569190X14000665

[19] Amesim help files, 2019.

[20] R. Burns, *Advanced control engineering*. Elsevier, 2001.

[21] G. Fernandez, C. Park, N. Kim, and R. Haftka, "Review of multi-fidelity models," vol. Fernández-Godino, M. Giselle, Chanyoung Park, Nam-Ho Kim, and Raphael T. Haftka. "Review of multi-fidelity models." arXiv preprint arXiv:1609.07196 (2016)., 2017.

[22] W. Keiper, A. Milde, and S. Volkwein, *Reduced-order modeling (ROM) for simulation and optimization: powerful algorithms as key enablers for scientific computing*. Springer, 2018.

[23] F. Donida, G. Ferretti, S. M. Savaresi, and M. Tanelli, "Object-oriented modelling and simulation of a motorcycle," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 14, no. 2, pp. 79–100, 2008. [Online]. Available: https://doi.org/10.1080/13873950701847090

[24] M. Tanelli, F. Schiavo, S. M. Savaresi, and G. Ferretti, "Object-oriented multibody motorcycle modelling for control systems prototyping," in *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, Conference Proceedings, pp. 2695–2700.

[25] D. Moreno, J. Kang, P. Talaia, M. Hajžman, L. Hyncik, and G. Evangeli, *Development of overall methodology for accident simulations*, 2008.

[26] Simcenter 3D help files, 2019.

[27] N. Cofelice, A. Toso, D. Moreno, S. Donders, J. De Cuyper, and H. Van der Auweraer, *Motorcycle Dynamics Simulation Including Realistic Rider Behavior*, 2011.

[28] E. Lopez and N. Nigro, "Validation of a 0d/1d computational code for the design of several kind of internal combustion engines," *Latin American applied research*, vol. 40, pp. 175–184, 2010.

[29] E. Haddad, D. Chalet, and P. Chesse, “Improved throttle valve modeling for spark-ignition engine simulations,” vol. 233, no. 6, pp. 1614–1622, 2019. [Online]. Available: https://journals.sagepub.com/doi/abs/10.1177/0954407018775823

[30] B. A. Thompson, “Development of a high-fidelity engine modeling framework in simulink with automated combustion parameter tuning,” Conference Proceedings.

[31] L. Guzzella and C. Onder, “Introduction to modeling and control of internal combustion engine systems,” 2004.

[32] W. Trindade and R. Santos, “Combustion modeling applied to engines using a 1d simulation code,” in *25th SAE BRASIL International Congress and Display*, 2016, Conference Paper. [Online]. Available: https://doi.org/10.4271/2016-36-0347

[33] J. Bohbot, M. Miche, P. Pacaud, and A. Benkenida, “Multiscale engine simulations using a coupling of 0-d/1-dmodel with a 3-d combustion code,” *Oil & Gas Science and Technology - Revue de l’IFP*, vol. 64, 2009.

[34] F. Lafossas, O. Colin, F. Le Berr, and P. Menegazzi, “Application of a new 1d combustion model to gasoline transient engine operation,” in *2005 SAE Brasil Fuels & Lubricants Meeting*, 2005, Conference Paper. [Online]. Available: https://doi.org/10.4271/2005-01-2107

[35] G. Alix, C. Pera, J. Bohbot, and A. Baldari, “Comparison of 0d and 1d duct system modeling for naturally aspirated spark ignition engines,” in *SAE International Powertrains, Fuels and Lubricants Meeting*, 2011, Conference Paper. [Online]. Available: https://doi.org/10.4271/2011-01-1898

[36] C. L. Huang and M.-C. Shih, “Design of a hydraulic anti-lock braking system (abs) for a motorcycle,” *Journal of Mechanical Science and Technology*, vol. 24, pp. 1141–1149, 2010.

[37] “Training - hyd2: Design of hydraulic systems and components,” Simcenter Amesim training files, 2018.

[38] N. Vasiliu, D. Vasiliu, C. Cãlinoiu, and R. Puhalschi, *Simulation of Fluid Power Systems with Simcenter Amesim*, 2018.

[39] “Training - hyd1: Introduction to hydraulic simulation,” Simcenter Amesim training files, 2020.

[40] N. Srivastava and I. Haque, “A review on belt and chain continuously variable transmissions (cvt): Dynamics and control,” *Mechanism and Machine Theory*, vol. 44, no. 1, pp. 19–41, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0094114X08001432

[41] Tony, “Aprilia mana 850 gt,” 2020. [Online]. Available: https://www.motorbikecatalog.com/curve/2013/26750/aprilia_na_mana_850_gt_abs.html

[42] A. Fraga, “Desarrollo de simulaciones en tiempo real de modelos multi-físicos para ensayos en banco de prueba de máquinas rotativas,” Mastersthesis, Universidade da Coruña, 2016.

[43] *Spare parts catalogue: NA 850 Mana GT*, Aprilia, 2009-2013.

[44] *KiTorq Rotor: Torque measuring unit (rotor) for a torque measuring flange*, Kistler, 2011.

[45] *Torque settings: Aprilia NA 850 Mana GT*, Aprilia, 2009-2013.

[46] S. Sitthiracha, S. Patumsawad, and S. Koetniyom, "An analytical model of spark ignition engine for performance prediction," 2006.

[47] L. Eriksson and L. Nielsen, *Modeling and control of engines and drivelines.* John Wiley & Sons, 2014.

[48] J. Heywood, *Internal Combustion Engine Fundamentals 2E.* McGraw-Hill Education, 2018. [Online]. Available: https://books.google.be/books?id=OmJUDwAAQBAJ

[49] R. Stone, *Introduction to Internal Combustion Engines.* Palgrave Macmillan, 2012. [Online]. Available: https://books.google.be/books?id=afYcBQAAQBAJ

[50] *Service station manual: Aprilia NA 850 Mana GT*, Aprilia, 2007.

[51] H. Nigus and E. Editor, "Kinematics and load formulation of engine crank mechanism," 01 2016.

[52] G. P. Blair, "Design and simulation of four-stroke engines," SAE Technical Paper, Tech. Rep., 1999.

[53] S. Narwade, C. Kumar, and S. Patil, "Multiple control parameters and functional mode considerations for gasoline ems engine control unit — a survey," 08 2017, pp. 1–5.

[54] R. Isermann, "Engine modeling and control," *Berlin: Springers Berlin Heidelberg*, vol. 1017, 2014.

[55] B. Birch and M. Amann, "Electronic fuel system development for air-cooled motorcycles," 09 2004.

[56] B. Ashok, S. Denis Ashok, and C. Ramesh Kumar, "A review on control system architecture of a si engine management system," *Annual Reviews in Control*, vol. 41, pp. 94 – 118, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578816300086

[57] R. M. Valle, J. E. M. Barros, J. G. C. Baeta, and F. J. P. Pujatti, "Mapping procedure applied to general engine management system for spark plug engines," in *SAE Technical Paper.* SAE International, 11 2004.

[58] P. Zal, "The catalog of motorbikes," 2020. [Online]. Available: https://www.motorbikecatalog.com/curve/2013/26750/aprilia_na_mana_850_gt_abs.html

[59] "Big-bang firing order," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Big-bang_firing_order

[60] T. Kimishima, T. Nakamura, and T. Suzuki, "The effects on motorcycle behavior of the moment of inertia of the crankshaft," *SAE Transactions*, vol. 106, pp. 1993–2003, 1997. [Online]. Available: http://www.jstor.org/stable/44731343

[61] P. B. U. Achi, "The design and testing of an automatic clutch," *Sadhana*, vol. 9, no. 3, pp. 233–238, 1986. [Online]. Available: https://doi.org/10.1007/BF02811966

[62] R. G. Weight, "High-torque capacity compliant centrifugal clutches," 2004.

[63] M. Milosavljevic, F. Berr, S. Laabidi, S. Venturi, F. Raymond, G. Zito, S. Magand, and P. Degeilh, “Innovative approach and tools to design future two-wheelers powertrain,” 11 2015.

[64] M. Tanelli, M. Corno, and S. Savaresi, *Modelling, Simulation and Control of Two-Wheeled Vehicles*, 01 2014.

[65] J. Meijaard and A. Popov, “Multi-body modelling and analysis into the non-linear behaviour of modern motorcycles,” *Proceedings of The Institution of Mechanical Engineers Part K-journal of Multi-body Dynamics - PROC INST MECH ENG K-J MUL D*, vol. 221, pp. 63–76, 03 2007.

[66] L. Leonelli and N. Mancinelli, “A multibody motorcycle model with rigid-ring tyres: formulation and validation,” *Vehicle System Dynamics*, vol. 53, no. 6, pp. 775–797, 2015. [Online]. Available: https://doi.org/10.1080/00423114.2015.1014820

[67] V. Cossalter, R. Lot, and F. Maggio, “A multibody code for motorcycle handling and stability analysis with validation and examples of application,” *SAE Technical Papers*, 09 2003.

[68] V. Cossalter, *Motorcycle Dynamics*. Vittore Cossalter, 2006. [Online]. Available: https://books.google.be/books?id=rJTQxITnkbgC

[69] C. Liu, “A multibody dynamics model of a motorcycle with a multi-link front suspension,” Ph.D. dissertation, 11 2017.

[70] H. B. Pacejka and E. Bakker, “The magic formula tyre model,” *Vehicle System Dynamics*, vol. 21, no. sup001, pp. 1–18, 1992. [Online]. Available: https://doi.org/10.1080/00423119208969994

[71] H. B. Pacejka, “Chapter 11 - motorcycle dynamics,” in *Tyre and Vehicle Dynamics (Second Edition)*, second edition ed., H. B. Pacejka, Ed. Oxford: Butterworth-Heinemann, 2006, pp. 517 – 585. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780750669184500113

[72] R. Sharp, S. Evangelou, and D. Limebeer, “Advances in the modelling of motorcycle dynamics,” *Multibody System Dynamics*, vol. 12, pp. 251–283, 10 2004.

[73] “Simrod experience: From belgian blocks to vehicle durability testing (part 2),” 2020. [Online]. Available: https://blogs.sw.siemens.com/simcenter/simrod-experience-from-belgian-blocks-to-vehicle-durability-testing-part-2/

[74] C.-Y. Lu and M.-C. Shih, “Application of the pacejka magic formula tyre model on a study of a hydraulic anti-lock braking system for a light motorcycle,” *Vehicle System Dynamics*, vol. 41, no. 6, pp. 431–448, 2004. [Online]. Available: https://doi.org/10.1080/00423110512331383848

[75] C. Canudas-de Wit, P. Tsiotras, E. Velenis, M. Basset, and G. Gissinger, “Dynamic friction models for road/tire longitudinal interaction,” *Vehicle System Dynamics - VEH SYST DYN*, vol. 39, pp. 189–226, 03 2003.

[76] K. Johanastrom and C. Canudas-de-Wit, “Revisiting the lugre friction model,” *IEEE Control Systems Magazine*, vol. 28, no. 6, pp. 101–114, 2008.

[77] M. Korayem, Y. Habibi Sooha, and Z. Rastegar, “Modeling and simulation of viscoelastic biological particles’ 3d manipulation using atomic force microscopy,” *Applied Physics A*, vol. 124, 04 2018.

[78] S. Jia and Q. Li, “Friction-induced vibration and noise on a brake system,” in *2013 IEEE International Conference on Information and Automation (ICIA)*, 2013, pp. 489–492.

[79] Tony, “Aprilia mana 850 gt,” 2020. [Online]. Available: https://www.motorbikecatalog.com/performance/2013/26750/aprilia_na_mana_850_gt_abs.html

[80] F. J. Laimböck and R. S. Kirchberger, “Development of a 150cc, 4-valve cvt engine for future emission and noise limits,” in *SAE Technical Paper*. SAE International, 09 1998. [Online]. Available: https://doi.org/10.4271/982052

[81] J. Watanabe, D. Arai, M. Tanaka, T. Abe, A. Ogasawara, M. Tsuchiya, and R. Tsubota, “Development of dual clutch transmission for large motorcycles,” *SAE International Journal of Engines*, vol. 4, pp. 2157–2165, 06 2010.