POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Model of a Tracked Vehicle for Agricultural Applications

Laurea Magistrale in Automation and Control Engineering - Ingegneria dell'Automazione

Author: Aleksandar Pavlovski

Advisor: Prof. Gianni Ferretti

Co-advisor: Luca Bascetta

Academic year: 2025-2026

1. Introduction

Tracked vehicles are widely used, particularly in agricultural applications, where they often operate on soft and deformable soil. The current trend is moving toward autonomous tracked vehicles, which makes accurate modeling and control of such systems of essential importance. This, in turn, requires a detailed dynamic model of the tracked vehicle and an appropriate model of the soil on which it operates.
In this work, a MATLAB/Simulink model of a tracked vehicle is developed based on [2], together with a soil model derived from the Bekker pressure–sinkage formulation. The proposed model is able to simulate 2D vehicle motion in real time and to retain the state of the terrain, allowing the analysis of scenarios in which the vehicle passes multiple times over the same area.

2. Tracked vehicle model

In this chapter, a brief description of the tracked vehicle model is presented. The model proposed in [2] is particularly suitable for MATLAB/Simulink implementation due to its simplicity and modular structure. It allows different dynamic effects to be implemented in separate functional blocks, which results in a clear and well-structured modular architecture. The model implemented for the purposes of this thesis takes the left and right sprocket torques as inputs and outputs the resulting vehicle dynamics. The main forces and torques, which are concentrated at specific points of the vehicle, are illustrated in the following figure.

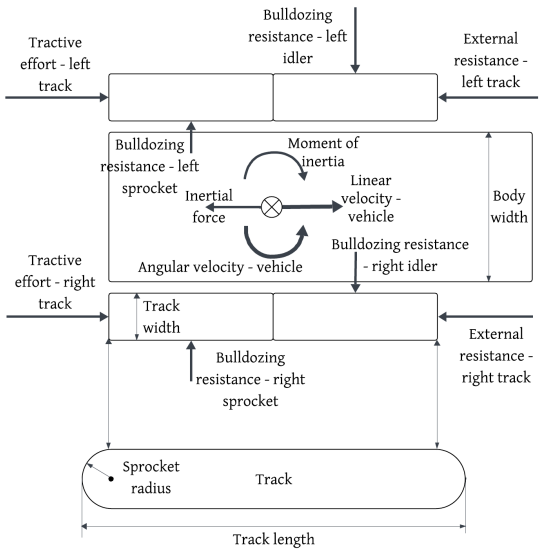

Figure 1: Moments and forces acting on the vehicle

The tractive efforts shown in Figure 1 are among the essential effects that must be estimated and are computed according to the formulation presented in [2]. The corresponding equation is reported below.

$$F = \operatorname{sign}(\tilde{j})\, bl \left[c + \frac{p_s + p_i}{2} \tan\phi \right] \times \left[1 + \frac{K}{2|\tilde{j}|} \left(e^{-2|\tilde{j}|/K} - 1 \right) \right] \mathbf{u} \tag{1}$$

In the above expression, $\tilde{j}$ denotes the mean shear displacement under the track, while b and l represent the track width and contact length, respectively. The parameters $\tan\phi$, c and K correspond to the shear strength parameters of the track–terrain interaction model originally proposed by Bekker [1]. Pressures under the sprocket and idler are denoted by p_s and p_i, respectively.

Another essential equation for describing the vehicle–terrain interaction is the shear displacement estimation, as presented in [2]. This represents one of the main effects studied and estimated in the model. The corresponding equation is reported below:

$$\frac{d\tilde{j}}{dt} = -\frac{2r}{l} |\omega_s| \tilde{j} + r\,\omega_s - v \tag{2}$$

In this equation, r denotes the sprocket radius, ω_s is the angular velocity of the sprocket, and v represents the longitudinal velocity of the track. The core approach in the Simulink implementation was to encapsulate all model equations inside MATLAB Function blocks, while leaving only the state integrators outside. Additionally, an initialization script is used to pre-compute all required parameters before running the simulation, in order to avoid unnecessary computational overhead during execution. During simulation studies, it was observed that the model tends to be stiff due to the number of simplifying assumptions introduced. However, with appropriate techniques and "tricks" in Simulink, the model can be implemented efficiently. Stiffness was reduced by approximating sign functions using the hyperbolic tangent and adjusting the steepness coefficient. Following scheme illustrates the main modules implemented in MATLAB/Simulink and the signals that interconnect them.

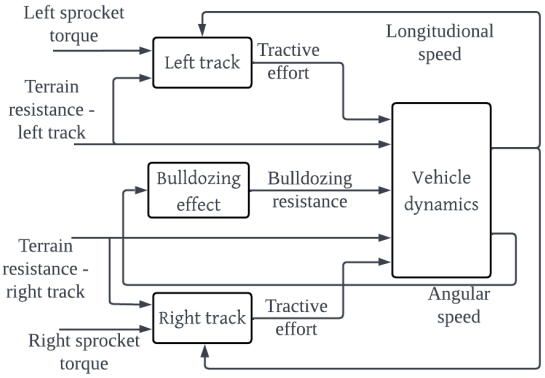

Figure 2: Main modules implemented in MATLAB/Simulink model of a vehicle

The model showed results very similar to those presented in [2], including oscillations in the shear displacement, especially when the torque was changed suddenly. A detailed analysis and discussion of these oscillations are provided in the thesis. Two types of oscillations were observed: numerical and model-related. Numerical oscillations arise from using the sign function and were reduced by approximating it with a hyperbolic tangent function. Model-related oscillations are caused by the strong coupling between applied torque, tractive effort, shear displacement, and vehicle motion. Using a first-order filter instead of a step input for the torque reduces these oscillations. This is also realistic, as no actuator can apply torque instantaneously.

3. Terrain model

The motivation for developing a detailed terrain model arises from the need to account for terrain state when a vehicle traverses the same path multiple times. In such cases, the soil is already compacted by previous passes. This is a realistic scenario in agricultural applications, where vehicles often move repetitively along predefined tracks.

In addition, the implementation of a three-dimensional vehicle model requires the ability to represent uneven terrain, such as small hills and furrows, over which the vehicle travels. For this reason, the terrain is discretized into small cells, and each cell implements the Bekker pressure–sinkage formulation.

3.1. Cell model

Figure 3 shows the main forces affecting the dynamics of a rigid body that represents a single terrain cell. The state of a cell is described by its sinkage z, which practically corresponds to the vertical displacement of the cell from its initial position under the influence of applied forces. When an external load p_{load} is applied, the cell generates a response pressure p based on the Bekker [1] pressure-sinkage model. The inertial force $m\ddot{z}$ and damping force $c'\dot{z}$ are also considered.

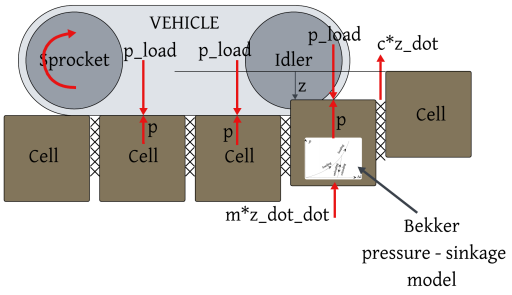

Figure 3: Rigid body dynamics of a cell

Based on mentioned effects, equation (3) was derrived.

$$\ddot{z} = \frac{1}{m_{\text{cell}}} \left[(p_{\text{load}} - p) A_{\text{cell}} - c'\dot{z} \right] \tag{3}$$

A simplification was introduced: the cell area A_{cell} (which would normally be used to compute forces from pressure) and the cell mass m_{cell} (used in the inertial term) are combined into a single coefficient τ_z in equation (4).

$$\ddot{z} = \frac{1}{\tau_z} \left(p_{\text{load}} - p - c''\dot{z} \right) \tag{4}$$

The MATLAB/Simulink implementation of a single terrain cell is presented in the following figure. Two main function blocks are used: one implements the Bekker pressure–sinkage model, while the other represents the cell dynamic equation (3). The model input is the applied load pressure, and the outputs are the cell sinkage and the resulting response pressure. The figure also shows multiple cells arranged in parallel, indicating that several cells operate simultaneously.

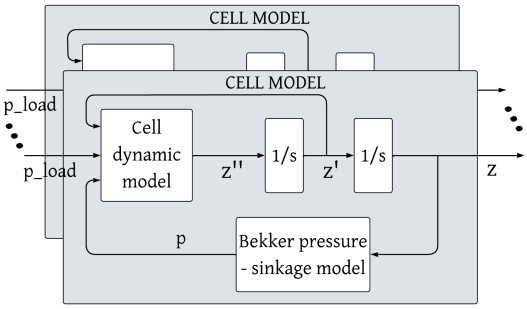

Figure 4: MATLAB/Simulink implementation of a cell

3.2. Parallel cell simulation challenge

To achieve a sufficiently detailed terrain representation, the terrain must be discretized into cells with small dimensions. In addition, in order to simulate realistic vehicle maneuvers, the overall terrain size must be adequately large. However, increasing the terrain dimensions while decreasing the cell size leads to a rapid growth in the number of cells and their associated states, making real-time simulation infeasible.
For example, a terrain of size 100 m × 100 m discretized into cells of 10 cm × 10 cm results in one million cells that would need to be simulated together with the vehicle model. To enable real-time simulation, the key idea adopted in this work is that only cells subjected to pressure are active, and only their states, namely the sinkage, evolve over time.

4. Terrain - vehicle interraction

4.1. Channels

A fixed and limited number of channels is predefined, where each channel is capable of simulating the dynamics of a single cell. Each active cell is assigned a channel, which is initialized with the states of that cell, and its states are updated through the assigned channel as long as the cell remains active. A dedicated block is introduced to handle channel assignment and to store the states of all terrain cells. This block, referred to as the `world_manager`, performs channel allocation efficiently, without the need to check for free channels, and uses a predefined channel

mapping for each cell. The logic of this block is described in the remainder of this work.

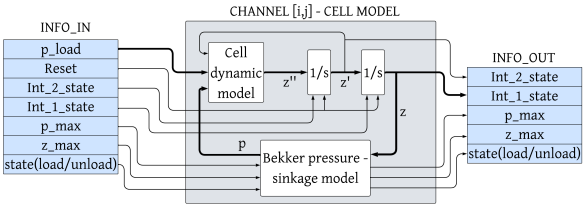

Figure 5: MATLAB/Simulink implementation of a channel

The term `P_load` denotes the load applied to the cell, while `int_1_state` and `int_2_state` represent the integrator states stored from the previous time the cell was active. These states are crucial for reinitializing the integrators whenever the cell becomes active again. The variable `int_1_state` represents the current sinkage z, while `int_2_state` corresponds to the first derivative of the sinkage.
A reset signal is sent to the integrators to indicate that new initial conditions must be taken from `INFO_IN`. The variable `state` indicates whether the cell was previously in the loading or unloading/reloading phase, which is essential for selecting the appropriate Bekker equation. Since the Bekker loading and unloading equations require the parameters `p_max` and `z_max`, which represent the maximum sustained pressure and maximum sinkage, respectively, these quantities are stored from the cell's previous activity and passed as part of the input structure.
The output data contains all the information that must be stored so that it can be reused the next time the same cell becomes active. It is important to note that a channel does not have knowledge of which terrain cell it is currently simulating; this mapping is handled entirely by the `world_manager` module.

4.2. World_manager block

The `world_manager` block is responsible for managing terrain data and coordinating the simulation by storing information for every cell within the `Cell[N,N]` matrix. It determines which cells are active based on the vehicle's pose and assigns each active cell to a specific channel within the `CHANNEL[N_channels, N_channels]` structure. When a cell becomes active, its stored parameters, such as `Int_1_state`, `Int_2_state`, `p_max`, `z_max`, and the current `state`, are sent via `INFO_IN` to initialize the assigned channel. This fixed channel matrix approach avoids dynamic memory allocation and computational overhead, as the `WORLD_MANAGER` uses a sector-based mapping strategy where each `Cell[i,j]` is directly linked to a corresponding channel. During each simulation step, the channel processes the interaction and returns updated data through `INFO_OUT`, which is then written back into the cell to maintain persistence. A cell remains active until the unloading phase is complete and the reaction pressure returns to zero.

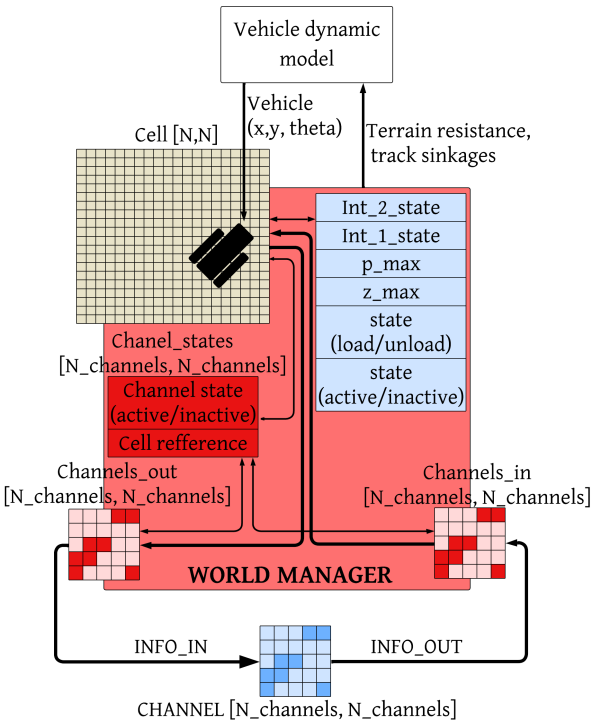

Figure 6: MATLAB/Simulink implementation of a `world_manager` block

This approach resolves the challenge of simulating a massive number of terrain states. A terrain that can contain millions of cells is simulated using a fixed number of approximately one hundred channels, with only a few dozen active at any given time. Consequently, instead of millions of parallel simulations, the model is reduced to just a dozen, enabling real-time simulation.

The following figure shows the sinkage stored in the terrain matrix after the vehicle has passed.

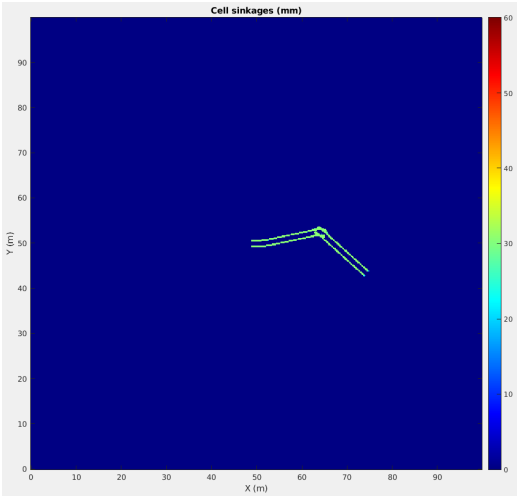

Figure 7: Terrain sinkage matrix

5. Simulation results

The model of the tracked vehicle in [2] was qualitatively validated by comparing simulation results with real-life experiments. A vehicle with given specifications was driven on a hard, homogeneous clayey terrain. One experiment consisted of applying a torque of 2000 Nm to both the left and right sprockets for 4 seconds, after which one of the sprockets was left idle while the other continued to receive 2000 Nm. The result of this experiment was that the vehicle continued to follow a straight trajectory with only a very small deviation. Simulation results presented in [2] are summarized as follows:

- From the moment of setting one sprocket idle, the vehicle covers approximately 20 m, while the lateral deviation is about 8 cm
- Sinkage values are approximately 1–2 mm, which corresponds to hard terrain conditions
- The yaw angle changes from 0° to 0.4°
- The shear displacement is in the range of 10 cm, and the idle sprocket exhibits negative shear
- The vehicle assumes a nose-up behaviour, meaning that the rear part of the vehicle, experiences greater sinkage

The following graphs present a comparison between the simulation results reported in [2] and the results obtained from the simulation developed in this thesis.

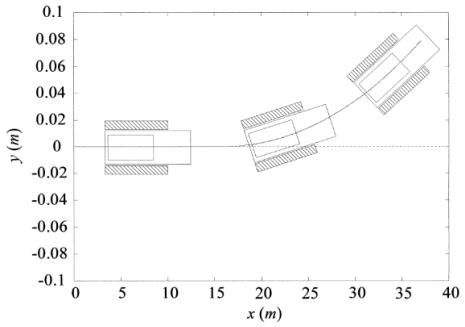

Figure 8: Vehicle position simulation from [2]

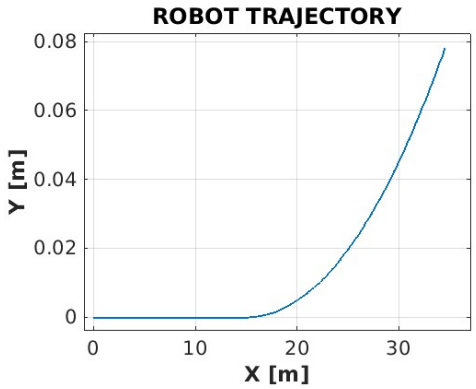

Figure 9: Vehicle position from this thesis model simulation

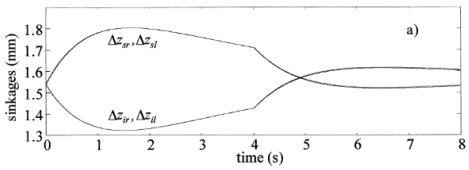

Figure 10: Average sprocket and idler sinkages simulation from [2]

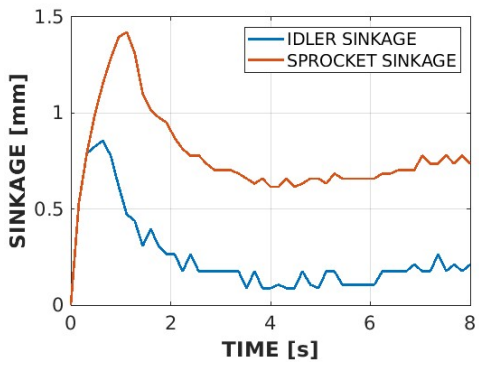

Figure 11: Average sprocket and idler sinkages from this thesis model simulation

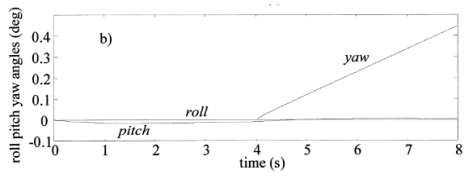

Figure 12: Yaw angle simulation from [2]

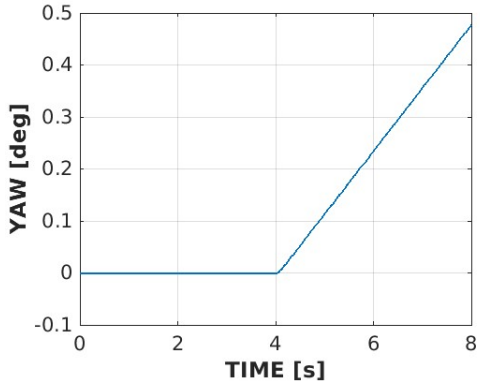

Figure 13: Yaw angle from this thesis model simulation

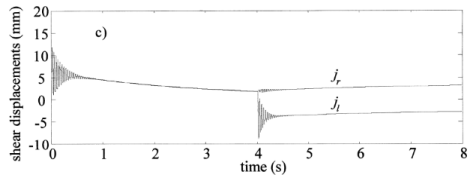

Figure 14: Shear displacements simulation from [2]

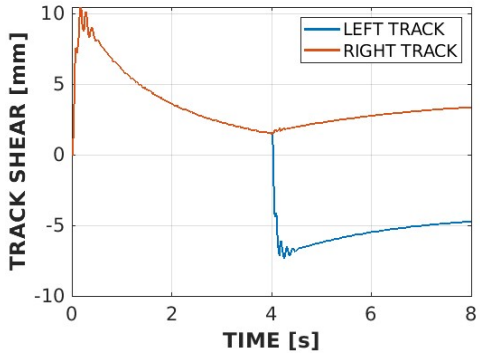

Figure 15: Shear displacements from this thesis model simulation

It can be observed that the simulation performed with the model developed in this thesis provides results quite similar to those presented in [2]. The vehicle follows a comparable path. Regarding sinkages, the nose-up behavior and the range of 1–2 mm are captured in the simulation of the model developed in this thesis. However, a different behavior is observed compared to the one reported in [2]. This discrepancy is most likely due to the fact that the model developed in this thesis is 2D, whereas the model in [2] is 3D. The yaw angle behavior in both simulations is almost identical. Shear displacements are in the range of 10 mm, and the idle track exhibits negative shear as expected. The model developed in this thesis demonstrates good agreement with the simulation from [2].

6. Conclusions

In this thesis, a 2D dynamic model capable of real-time tracked vehicle and terrain simulation was developed. A key contribution of the work is the efficient management of terrain states through a channel-based allocation strategy. This approach resolved the challenge of computational overload by reducing the simulation space from millions of potential cell states to a fixed set of active channels, thus enabling real-time performance without sacrificing physical accuracy. The integration of the Bekker pressure-sinkage formulation proved to be an effective yet computationally simple method for approximating soil behavior. Furthermore, the preservation of terrain states such as sinkages allows for the simulation of multiple passes over the same path, providing a solid foundation for extending this model to 3D.

7. Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Gianni Ferretti, and my co-advisor, Luca Bascetta, for their constant availability and insightful guidance, which invariably steered this research in the right direction.

References

[1] Bekker. Introduction to terrain-vehicle systems. 1969.

[2] Roberto Girelli Gianni Ferretti. Modelling and simulation of an agricultural tracked vehicle. *Journal of Terramechanics 36 (1999) 139 - 158*, pages 140–158, 1999.