MULTIBODY DYNAMICS 2005, ECCOMAS Thematic Conference
J.M. Goicolea, J.Cuadrado, J.C.García Orden (eds.)
Madrid, Spain, 21–24 June 2005

EVALUATION OF DIFFERENT NUMERICAL INTEGRATORS APPLIED TO A FORKLIFT TRUCK REAL-TIME SIMULATOR

Xabier Iriarte[*], Javier Gil[*], Jorge San Miguel[*], Jesús Calleja[†], Jesús Mª Pintor[*], Jose Manuel Jiménez[†]

[*] Applied and Computational Mechanical Engineering
Public University of Navarra
Campus Arrosadia s/n, 31006 Navarra, Spain
e-mail: `iriarte.28842@e.unavarra.es`, web page: `http://www.imem.unavarra.es`

[†] STT Ingeniería y Sistemas, S.L.
Parque Empresarial Zuatzu, 2nd Floor, Ed. Easo, 20018 San Sebastian, Spain
e-mails: `jcalleja,jmjimenez@stt.es`

Keywords: Simulation, Multibody, Dynamics, Forklift Truck, Numerical Integration

Abstract. *In this article a forklift truck model is presented, which has been developed for a real-time simulator with man-in-the-loop. The simulator has been developed in order to prevent the industrial accidents derived from the use of forklift trucks. The simulator will be used by the operators who are trying to obtain the forklift truck driving license. This way, the operator can train some situations such as load dropping, sliding or overturning that cannot be trained in a real forklift truck driving session. The simulator consists of different software components: the 3D scene environment, the graphic engine, the forklift truck dynamic behavior model, the numerical integration module, the collision module, the driver evaluation and the simulator manager. The efficiency of overall software implementation is a very important matter that decides whether the simulator is of good quality or not. In this article a precise dynamic model presentation is made: the driven degrees of freedom of the model, the change of parameter values in execution time, the ground to wheel forces, the braking torque modeling and the "magic formulae" parameters calculation are explained. Moreover, the approach to the mechanical equations is explained and also some numerical experiments are done with the model. Finally, some results are drawn from the numerical experiments, both about the dynamic behaviour of the model and some numerical integrators' computational time. The results of the numerical experiments have helped us to draw conclusions about the numerical integration method to be used as well as how it fulfils the requirements of a real-time application. These results also have confimed that the model works properly and behaves as expected.*

1 INTRODUCTION

In this article a forklift truck real-time simulator is presented and some integration methods are used in order to solve the differential equations that represent the dynamic behaviour of the forklift truck. This project, called SIMUCEL, has been developed by the ACME research group (Applied Computational and Mechanical Engineering) of the Public University of Navarra. The aim of this project is to reduce the number of industrial accidents related to the use of forklift trucks. This way, the tool developed trains coming drivers in the correct use of these vehicles focusing on the prevention of industrial accidents related to driving forklift trucks.

Some of the advantages of a tool like a real-time simulator are obvious. The user of the application can train some situations impossible to train in real conditions such as: dropping or sliding of the load, and sliding or overturning of the forklift truck itself. Moreover, the simulator can not only represent those situations without risk, but also evaluate the behaviour of the driver, improving his/her skills and making him/her get into the habits of driving properly. This will certainly lead to a smaller probability of accident; and that is the aim of the simulator.

There is another attractive application for a simulator like ours. This simulator can be perfectly used for evaluating the people that want get the forklift truck driving license. Since the simulator can measure and evaluate the manoeuvres performed by the driver, the evaluator has a completely objective tool to evaluate the candidate's skills and determine whether the candidate passes or fails the exam.

The simulator consists of some software modules. Each of them does its job while all of them interact.

- Menu Structure
- Virtual Scene
- Graphic Engine
- Dynamic Model
- Collision Calculation Module
- Simulation Engine
- Driver Evaluation Module

The Menu Structure is an application developed for the user to choose the characteristics of the simulation environment in which he/she is going to train. Moreover, this Menu Structure manages all the information of each pupil and his/her performance in all the training sessions as well as the information about his/her theoretical and practical exams.

The Virtual Scene consists of a conventional warehouse in which the most common elements have been introduced such as: shelves, palets, obstacle, doors, docks and trucks.

The Graphic Engine used, which manages all the information related to the representation of the Virtual Scene, is called OGRE: (Object-oriented Graphic Rendering Engine). It is free software and can be downloaded from the Internet.

The forklift truck Dynamic Model has 8 degrees of freedom and 4 driven coordinates. It makes it possible to represent the dynamic behaviour of the forklift truck in order to develop a realistic simulator.

The Collision Calculation Module checks the existence of collisions between objects in the scene. It makes it possible to know if any collision between bodies is happening. All the possible collisions between the forklift truck, palets, walls, docks, trucks, cones and shelves are taken into account. This module is tightly related to the Driver Evaluation Module that evaluates whether the performance of the driver is good or not. Every mistake will be associated with a worrying level and finally the mark of the exam is shown.

And last but not least, The Simulator Engine is in charge of managing the information that all the other modules have to share. It is in charge of taking the peripherals' values, introducing them into the dynamic model, taking the output values of the model, and sending them both to the Graphical Engine and to the Collision Calculation Module, so that they can represent the Virtual Scene and calculate if there is any collision in that certain moment.

In the present article we are going to focus on exposing in detail all the characteristics of the dynamic model of the forklift truck, and the way the equations of motion are built. Moreover, a comparison between some numerical integrators is made in order to test their suitability to integrate the dynamic equations of motion in a real-time environment simulator.

2 CHARACTERISTICS OF THE CHOSEN FORKLIFT TRUCK

There exist many types of forklift trucks in the market that are used for manipulating loads in warehouses. However, only few of them are used in the average enterprise. The forklift truck that has been chosen as the one from which to build a model is a Linde E16c. This is a 3200 kg tricycle vehicle that lifts up to 1600 kg. The rear wheel is the driver one while the other two are the force driving ones.

3 DYNAMIC MODEL

The dynamic model of the forklift truck has been developed with the 6 solids it consist of: the chassis, the three wheels, the mast and the pitchfork. The coordinates selected for the developement of the model have been chosen independent from each other. Since the forklift truck is an open chain mechanism, and all the joints being of a single relative movement, using independent and relative coordinates seemed to be the most suitable selection in order to integrate the model in real-time. The chosen coordinates are the following:

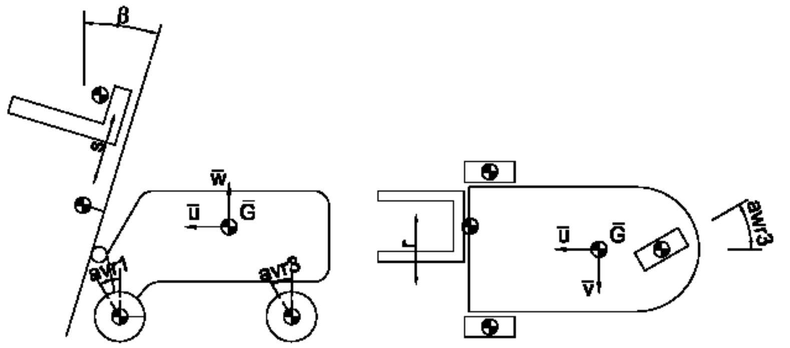


Figure (1)

Three angles for the orientation of the chassis:

- an Euler angle in the W direction awc

- two Euler angles in the U (auc) and V (avc) directions. The assumption of small rotations is done in order to linearise two of the coordinates of the model and make it lighter for the numerical integration.

Three coordinates for the position of the chassis

- x, y and z coordinates to position the centre of gravity of the chassis

Three coordinates for the three wheels' rotation around their own axis

- they have been called avr1, avr2 and avr3 respectively

An extra rotation angle around the vertical (W) axis in order to drive the rear wheel (awr3)

And finally, the three coordinates that drive the mast and the pitchfork.

- the beta angle for the rotation of the mast relative to the chassis in the V (lateral) direction. As in with the auc and avc, the assumption of small rotations is done for this beta angle.

- the s and r translation coordinates that represent the planar joint between the pitchfork and the plane of the mast. These are the coordinates that make possible to lift the load and move it laterally while holding it.

As we have already mentioned, all these coordinates are independent. Thus, there will not be any algebraic equation which relates the coordinates with each other. Nevertheless, not all of the coordinates will be treated as Dynamic Degrees Of Freedom (DDOF) and this will lead to some constrain-like equations. The awr3, beta, s and r coordinates are going to be the kinematically driven ones, and we will refer to them as the Driven Coordinates (DC) while the rest will be called DDOF. This way, our forklift truck model will consist of 13 generalised coordinates from which 4 will be DC and the other 9 will be DDOF.

4 THE DRIVEN COORDINATES (DC)

In the present model, some of the DOF-s, the so called DC, have been driven using constrain equations as the following: $q_i - f_i(t) = 0$. This way, and taking into account that there are no constrain equations, (algebraic equations involving coordinates), the Jacobian matrix obtained from deriving the driving constrain equations leads as follows: $J = [I \quad 0]$.

When we build the differential equations of motion, whatever the formulation used is (Lagrange, Euler, Penalty formulation, ...), it is necessary to evaluate the position, velocity and acceleration functions of the DC. This would not be a problem if we knew those values for any time. But that is not the case. Actually, since this model has been developed for a real-time environment with man-in-the-loop, not even the position of the DC is known in advance. That is the reason why the information of the position of the DC has to be taken from the peripherals and their first and second derivatives have to be calculated with any numerical derivative formula each time step. In fact, only the rear wheel's orientation is going to be driven by position making it necessary to calculate the two derivatives numerically. The other three DC, beta, r and s, are going to be driven by velocity. That is, the value taken from the input peripherals is going to represent the velocity of those coordinates and the position and acceleration will have to be calculated by integration for the first and differentiation for the second. The election of the right derivation and integration formulae could be crucial when solving some dynamic multibody systems.

Even if the effect of the reaction forces may not be relevant, suitable differentiation and integration formulae have to be chosen in order to keep these forces under control.

5 CHANGE OF PARAMETER VALUES IN EXECUTION TIME

The dynamic behaviour of any forklift truck is in general quite slow. But this does not mean that the dynamic forces will not be able to make the forklift truck overturn. The most dangerous situation in which the risk of overturning exists happens when the pitchfork holds a load at a high height, (with a big value for the s coordinate). In this situation, any acceleration in any of the plane directions leads to a very big torque around the perpendicular axis to the direction. The mass of the load is big and the distance from the load's centre of gravity to the forklift's centre of gravity is also big. That is why even for low accelerations, the dynamic loads are so important.

In order to simulate this phenomenon, it was necessary to develop an algorithm where the mass of the pitchfork was changeable. Moreover, since any type of load could be caught, the algorithm had to be able to change the mass of the pitchfork to any value. The Simulation Engine Module is in charge of this task and will change the right parameters when a load is caught or left. Since the coordinates are independent and the mass matrices will vary with respect to time, the mass matrix of the pitchfork, as all the other, will be evaluated each time step. Thus, changins some parameter values in execution time does not add extra work. This change of mass could lead to instability while solving the equations of motion, but this problem will not occur in our model since all the mass-value changes are going to be done when all the velocities and accelerations of the model are negligible. This first model approximation will assume that when the load is caught, it will be attached to the pitchfork and will not slide with respect to the pitchfork. In future releases of the model, the acceleration of the load will be calculated every time-step in order to warn the user that the load is sliding and will certainly drop.

6 DRIVING TORQUE DISTRIBUTION

This time the way of modelling the action of the accelerator has not been driving any of the coordinates of the model. If we wanted to do so, we should suppose not only that all the wheels roll without sliding, but also that we were able to drive the rotation of the two forward wheels. Even if we did so, a model built like that would not represent the real dynamic behaviour of the forklift truck. That is the reason why a model in which the forklift truck moves due to some external forces has been developed.

In this case, an external torque has been exerted to the two driving wheels, and the reaction torque has been applied to the chassis. The quantity of torque exerted to the wheels depends on what the user decides every time step through the input peripherals. However, the torque that the engine applies to the wheels is not always half of the total for each.

The way the torque is divided in a common car mainly depends on the kinematics and dynamics of the balance gear, and also on the different external forces applied to each of the wheels. But this is not the same for a forklift truck. On the contrary, what really happens in a vehicle like ours is that the torque applied to each of the driving wheels is generated by two different electrical engines, and there exist an electronic control system that is in charge of distributing the torque for each wheel in terms of the total torque exerted and the driving angle of the rear wheel. Taking into account this phenomenon but without any information about the actual torque control criteria, a torque distribution law has been proposed under some reasonable criteria:

- The torque for each wheel is the same for a zero rear wheel driving angle $$\left[awr3 = 0 \rightarrow M_1 = M_2 = \frac{M}{2} \right] \tag{1}$$
- When the instantaneous rotation centre is situated on a certain wheel, the torque exerted to that is zero $$\left[awr3 = \arctan(\frac{2 \cdot L}{H}) \rightarrow M_1 = M; M_2 = 0 \right] \tag{2}$$
- When the angle of the rear wheel equals $\pm 90^{\circ}$, the instantaneous rotation centre is situated in the middle point between the two forward wheels; and in this case the torque applied to both wheels is the same although the signs are opposite. $$\left[awr3 = \pm 90^{\circ} \rightarrow M_1 = -M_2 \right] \tag{3}$$
- The sum of the absolute values of the two torques is always equal to the value of the total torque that the user of the simulator decides. $$\left[|M_1| + |M_2| = M \right] \tag{4}$$

These four statements lead to two torque distribution functions in terms of the driving angle of the rear wheel (awr3) and the total torque that the user exerts (M):

$$
\begin{array}{ll}
\varphi < -\varphi_c \rightarrow M_1 = \dfrac{M}{(\pi - 2\varphi_c)} \cdot (\varphi + \varphi_c) & \varphi < -\varphi_c \rightarrow M_2 = \dfrac{M}{(\pi - 2\varphi_c)} \cdot (\varphi + \pi - \varphi_c) \\
-\varphi_c < \varphi < \varphi_c \rightarrow M_1 = \dfrac{M}{2\varphi_c} \cdot (\varphi + \varphi_c) & -\varphi_c < \varphi < \varphi_c \rightarrow M_2 = \dfrac{M}{2\varphi_c} \cdot (\varphi_c - \varphi) \\
\varphi_c < \varphi \rightarrow M_1 = \dfrac{M}{\pi - 2\varphi_c} \cdot (\pi - \varphi - \varphi_c) & \varphi_c < \varphi \rightarrow M_2 = \dfrac{M}{\pi - 2\varphi_c} \cdot (\varphi_c - \varphi)
\end{array}
\qquad (5)
$$

$$\boxed{\varphi = awr3}$$

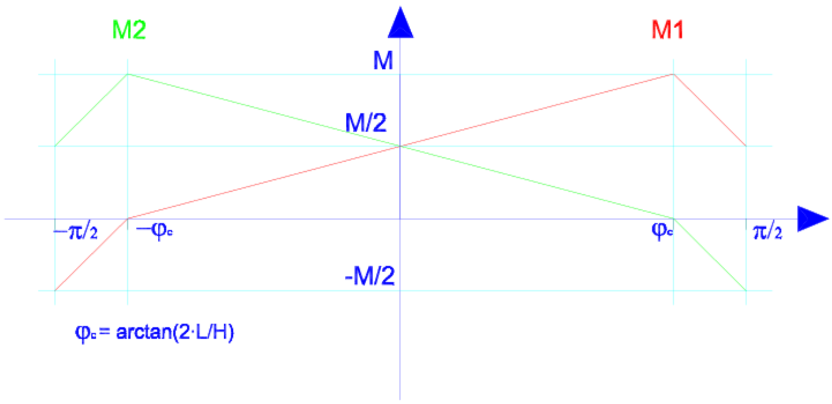


Figure (2)

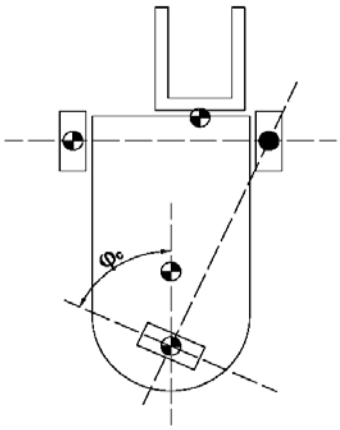


Figure (3)

The torque exerted to a wheel is also weighed up by the ratio of the total Fz force exerted to that wheel.

7 THE GROUND TO WHEEL FORCES

As mentioned above, we have chosen a force driven model in order to represent the way the forklift truck moves forward, turns and holds on the ground. Thus, the solution obtained will be much more like the real behaviour of the forklift truck, rather than the one obtained assuming non-sliding rolling wheels. The model that has been built for representing the forces that the ground exerts on the wheel is very much like the "Magic Formula" of Pacejka [2] and other authors, while ours is much more simple than those. Some simplifications have been made in order to build a light model that works, and represents properly the dynamic behaviour of the forklift truck.

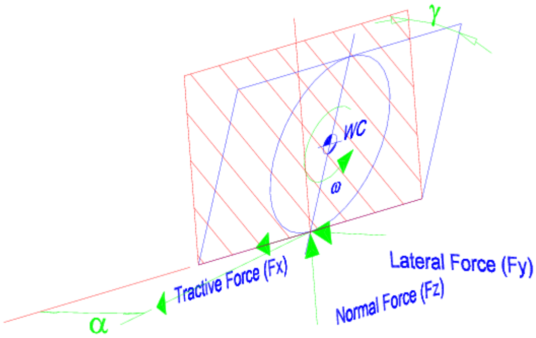


Figure (4)

The general analysis of the actual forces is going to be done and afterwards, some suitable simplifications are going to be made. Let us think of the general state for a wheel [2] which is represented in the figure 4:

The general momentum and force that the ground is going to exert to the wheel can be expressed in the reference system of the wheel itself. This way we can talk about Fx, Fy, Fz, Mx, My, Mz and also the projection of the velocity of the wheel's centre of gravity in the three directions. This force and momentum depend on many factors although we are going to calculate them in terms of only a few of them.

Since these vehicles do not have any kind of suspension, the pitch and roll angles (auc and avc) are going to take negligible values. Thus, the camber angle value (γ) of any of the wheels is going to be negligible (this has been taken into account using small angles for these rotations and linearising the problem for these variables). This is the reason why we can afford to vanish the momentum in the U direction associated with the camber angle (γ) of the wheel.

There exists an aligning momentum in the vertical (W) direction that will always try to reduce the rotational velocity of the wheel with respect to the ground in that vertical direction. Even if this momentum could be important not only for the dynamic movement of the vehicle but also for the stability of equations, we have vanished it because the arm for the force that generates the momentum is very small (smaller than the wheel's radius).

If we did an analysis of the vertical force distribution in the whole contact area (between wheel and ground) we would realise that the vertical force generates a momentum around the lateral (V) axis. This momentum appears against the moving direction and thus, it could be treated as friction force against the movement. Treating it like that, the application law for this momentum will be proportional to the derivative of the rotation of the wheel around its V axis and in the opposite direction:

$$M_y = -k \cdot \dot{\theta}_i \tag{6}$$

There are two more simplifications that are going to be made, and both of them are related to the functions that show the relation between the forces (lateral and longitudinal) and parameters they depend on (sliding and actual velocity of the wheel). But these are going to be explained in the definition of each of the three forces.

7.1 F_Z force:

The reason to explain this force first is that the other two (F_X and F_Y) depend on it. The aim of F_Z is to hold the whole forklift truck on the ground. Even if the forces expression is based on a spring-damper system, the position and velocity of the wheel's centre of gravity in the Z direction is not going to be important for us. We will only make use of it to calculate suitable forces to maintain the forklift truck on the ground.

The equation for the F_Z force is the following:

$$F_Z = K \cdot \left(Z - Z_0 - Z_{ground}(t) \right) - C \cdot \left(\dot{Z} - \dot{Z}_{ground}(t) \right) \tag{7}$$

where K is the stiffness of the "virtual spring", Z is the wheel's centre of gravity coordinate, Z_0 is the relaxed length of the "virtual spring", $Z_{ground}(t)$ is the height of the floor depending on time, C is the damping constant of the "virtual damper", $\dot{Z}$ is the derivative with respect to time of the Z coordinate and $\dot{Z}_{ground}(t)$ is the derivative of $Z_{ground}(t)$ with respect to time.

Even though the $Z_{ground}(t)$ and $\dot{Z}_{ground}(t)$ should depend on space coordinates rather than on time, in this first model it is not implemented like that. This way, we can perform simulations with any kind of slope but we can not implement that in the full real-time man-in-the-loop simulator since the directional derivatives of the ground height with respect to the moving direction are not being calculated yet.

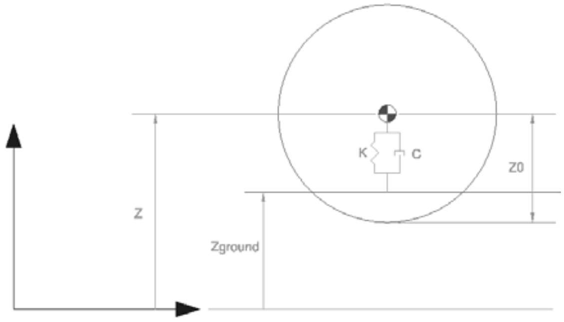


Figure (5)

As can be deduced from the equation (7), there will exist a holding force proportional to the remainder of substracting $Z_0 + Z_{ground}(t)$ to the Z coordinate. The other term of the equation will contribute to the overall force by adding a force proportional to $\dot{Z}$ and in the opposite direction to it.

The whole F_Z force will be zero if $Z > Z_0$; this would mean that the wheel does not rest on the ground, and so no force should be exerted.

This F_Z force, as the other two, is applied in the geometric contact point between the ground and the wheel, and is applied to the wheel rather than directly to the chassis. A suitable choice of the parameters will be exposed later on.

7.2 F_x and F_y forces:

Both the F_x and the F_y force magnitudes depend on the velocity direction of the wheel with respect to the ground and on the orientation of the wheel itself.

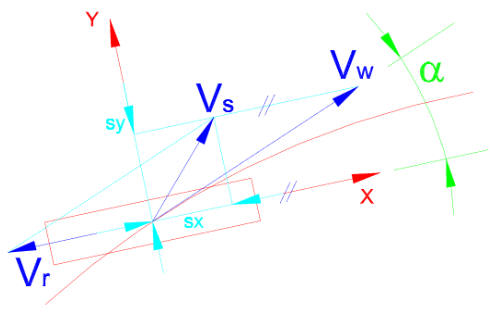


Figure (6)

We have decided to calculate both of them in the following way. The velocity vector of the geometrical centre of the wheel (V_W) is calculated first. Then the velocity vector of the contact point that belongs to the wheel with respect to the centre of it is calculated (V_r). Substracting the second to the first, we obtain the absolute velocity of the geometrical contact point as belonging to the wheel. We will call it V_S. In a perfect rolling without sliding situation, this velocity vector would be zero, but in this sliding model we will always assume that the forces will appear due to this slide, so it will appear almost every time step. Once this vector is calculated, its projections in the two in plane axis of the wheel must be calculated. We will call them sx and sy respectively. They are going to be scalar magnitudes so that we can use them in a scalar function.

The equation for the F_x force calculation is the following:

$$\begin{aligned} &F_X < -F_{x\max} \rightarrow F_X = -F_{x\max} \\ &-F_{x\max} < F_X < F_{x\max} \rightarrow F_X = -F_Z \cdot k \cdot s_X \\ &F_X > F_{x\max} \rightarrow F_X = F_{x\max} \end{aligned} \tag{8}$$

Where the k represents a proportional constant, s_X is the sliding in the X direction and $F_{x\max}$ is the maximum value for F_X. The negative sign of the central equation in (8 leads to a force in the opposite direction to the V_S vector's projection on the X axis (s_X). The k and $F_{x\max}$ constants will be given a value later on.

This function is very much like the functions shown by Pacejka in [2] for the F_X force. We have not used one of those functions because our definition of slip ratio is quite different from his'. However, the F_X functions shown in [2] help very much understanding the actual behaviour of a tyre. Our function for F_X written above is based on those general behaviour characteristics. The procedure for calculating F_Y leads as follows:

Whereas the F_X force depends on the modulus of the sliding in the X direction, the F_Y depends not only on the modulus of the sliding in the Y direction but also on the angle between the V_W direction and the wheel's rolling (X) direction. This angle is calculated depending on the projection of V_W on X and Y:

$$\alpha = \arctan\left(\left|\frac{V_{wy}}{V_{wx}}\right|\right) \tag{9}$$

The reason why we use the absolute value of the velocities quotient instead of the quotient with its sign itself is to avoid the situation in which the forklift truck is moving backwards, making the angle ±180º. The angle will actually be constant, but as the arctan function is defined to give values between -180º and 180º, any numerical negligible deviation from that aligned angle will give alternative 180 and -180 values. This phenomenon makes the simulation be very unstable and must be avoided.

Calculating the angle with the absolute value of the quotient, the sign of α is lost. We will solve this problem by giving α the sign of the projection of V_W in the lateral axis (V_{WY}). Once we get α, the F_Y will be calculated as follows:

$$\begin{aligned} &\alpha < -\alpha_0 \rightarrow F_Y = \mu \cdot F_Z \\ &-\alpha_0 < \alpha < \alpha_0 \rightarrow F_Y = -\mu \cdot F_Z \cdot \sin\left(\frac{\pi \cdot \alpha}{2\alpha_0}\right) \\ &\alpha_0 < \alpha \rightarrow F_Y = -\mu \cdot F_Z \end{aligned} \tag{10}$$

The equation (10) is represented in the following figure (7):

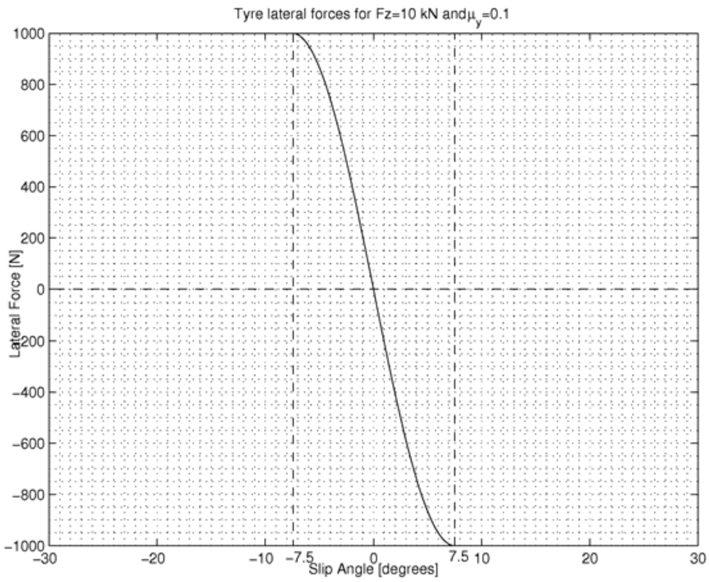


Figure (7)

8 BRAKING FORCE MODELING

The braking force has been modeled as a torque exerted to the wheels in the opposite direction to the wheel's movement. This is the easiest way of modeling the braking force, although some characteristics of it have to be taken into account.

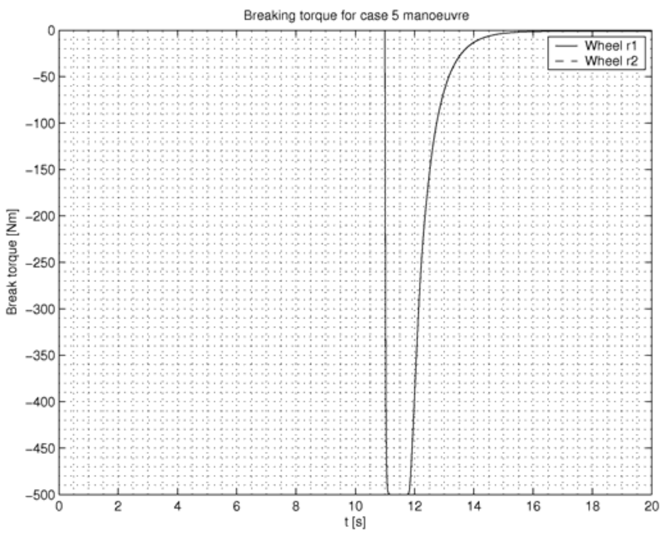


Figure (8)

If the braking torque were proportional to a constant input peripheral signal, the forklift truck would decrease its velocity and start moving backwards after stopping. Moreover, since it started moving backwards, the torque would change its sign to push the forklift forward, and it would stop and start moving forward once again. Thus, this schem would lead an unstable situation. This problem has been solved by multiplying the input peripheral signal by an exponential-like function which equals zero when the velocity of the wheel equals zero, and equals unity when the velocity is bigger than a certain velocity value. This way, when the forklift truck finally stops, no brake torque is exerted any more.

But this is not the only characteristic to take into account. When the user presses the brake sharply, the input signal tends to be discontinuous; and this does not help the model behave properly. In order to avoid the problem, a first order system has been introduced between the input signal and the exponential like function. This way, the torque derivative does not exceed an upper value and stability is achived.

So, the braking force has been basically modeled as a torque proportional to an input signal, while some changes have been done in order to maintain stability.

9 MAGIC FORMULAE PARAMETERS CALCULATION

9.1 F_Z parameters:

The parameters related to this force have been calculated in order to hold the forklift truck on the ground, and make the position and velocity variables keep stable.

This way, the K parameter has been chosen so that the natural frequency of the whole system was nearly 5 Hz. As the mass of the forklift is well known data, we can calculate the K in terms of the mass. The stiffness value obtained is: K=1.2E+6 N/m

The C parameter has been chosen in a very similar way. In order to achive stability, the value of ¿0.3? is given to the relative damping (ξ). And since we know the K and ξ, the calculation of C leads C=3.3E+3 N·s/m

The Z_0 parameter has been given the value of the radius of the wheel. This way, if the condition $Z > Z_0$ holds, it will mean that the wheel is not set on the ground.

9.2 F_X parameters:

The k parameter related to this force has been calculated so that the forklift truck accelerates in a numerically stable way. Once this goal was achived, it was also important to use a parameter value that did not make the equations system numerically stiff. Moreover, as no forklift truck can make its forward wheels slide if it is not in a braking manoeuvre, the maximum F_X force parameter ($F_{x\max}$) has been chosen in order to slip (reach the $F_{x\max}$ force) in a sharp braking, but not at an acceleration manoeuvre.

The values finally chosen are the following:

$$k = 0.25 \qquad F_{x\max} = 4500N$$

9.3 F_Y parameters:

This time, since our definition of slip angle is the same as the one in [2], we have decided to use the functions shown in [2]. However, there are not function equations in [2] but only graphical representations of them, and so, we have designed some functions that would nearly fit the graphics of the book. The functions' equations have been written in section 7: "The ground to wheel forces". The values for the parameters that would make our equation fit the graphics shown in [2] are:

$$\mu = 0.1 \qquad\qquad \alpha_0 = 7.5^{\circ}$$

10 APPROACH TO THE MECHANICAL EQUATIONS OF MOTION

In order to approach the mechanical equations of motion, the 3D_Mec-Matlab software library has been used. 3D_Mec-Matlab [4, 5, 6] is a pre-processor based on the MatLab Symbolic Toolbox which allows developing equations of motion of mechanical systems in a symbolic form.

The mechanical problem is conceptually divided into two sections:

- Kinematic description, involving geometric parameters and generalized coordinate definition. The symbolic-based kernel of the library allows the user to define any kind of coordinate. The model description can be carried out with the aid of a symbolic kernel which provides a user-friendly Cartesian interface including kinematic structures such as Points, Projection Basis and References. Required kinematic quantities such as positions, velocities and accelerations are automatically calculated by the library symbolic kernel. Last step in the kinematic description is the constraint definition, which can be made in both, matematical and Cartesian way.

- Dynamic description, involving inertial and external load definition. The inertial properties of the model are introduced with the aid of a dynamic Cartesian interface which includes structures such as Gravity Centre Positions, Inertia Tensors and Body References. On the other hand, external loads are introduced in a Cartesian way, that means, the user can directly define external forces and torques. Required dynamic quantities such as Mass Matrices and Load Vectors are symbolically calculated and exported in a MatLab-function form. The library kernel allows the user to obtain equations of motion for both Lagrange and Newton-Euler formalisms.

Thinking about real-time dynamics, Lagrange equations seem to be the more suitable ones. The model is represented by using 13 lagrangian (relative) coordinates. In order to reduce the symbolical expression of Mass Matrix and Load Vector, 3 rotation coordinates –avc, auc and beta- are linearized from its kinematic definition. As said before, 4 driven degrees of freedom –awr3, beta, r and s- are defined in the form:

$$\mathbf{q}_d = \mathbf{z} \tag{11}$$

Bearing in mind the Lagrange formalism, the equations of motion take the form

$$\begin{bmatrix} \mathbf{M}_{ii} & \mathbf{M}_{id} & \mathbf{0} \\ \mathbf{M}_{di} & \mathbf{M}_{dd} & \mathbf{I}_4 \\ \mathbf{0} & \mathbf{I}_4 & 0 \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}}_i \\ \ddot{\mathbf{q}}_d \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q}_i \\ \mathbf{Q}_d \\ \ddot{\mathbf{z}} \end{Bmatrix} \tag{12}$$

where $\ddot{\mathbf{q}}_i$ denotes the dynamic acceleration (DDOF) vector while $\ddot{\mathbf{q}}_d$ is the driven acceleration (DC) vector and $\mathbf{Q}_d = \mathbf{0}$ because no external loads are applied on DC-type coordinates.

Driven Accelerations, $\ddot{\mathbf{q}}_i$, can be solved from the first set of uncoupled equations of system (11) as

$$\ddot{\mathbf{q}}_i = [\mathbf{M}_{ii}]^{-1}(\mathbf{Q}_i - \mathbf{M}_{id}\ \ddot{\mathbf{z}}) \tag{13}$$

where Mass Matrices $\mathbf{M}_{ii} \in (9,9)$ and $\mathbf{M}_{id} \in (9,4)$ are state-dependent and must be evaluated every time-step.

Moreover, solution for Lagrange multipliers, $\boldsymbol{\lambda}$, if required, can be obtained from the second set of equations as

$$\boldsymbol{\lambda} = \mathbf{Q}_d - (\mathbf{M}_{di}\ \ddot{\mathbf{q}}_i + \mathbf{M}_{dd}\ \ddot{\mathbf{z}}) \tag{14}$$

where λ_j has the physical sense of the *driven generalized force* that acts over the corresponding *driven coordinates.*

The numerical integration of the dynamic model can be done with the aid of some numerical interface functions. Previous to any simulation, the user should take a look to initialisation files dealing with (kinematic and dynamic) parameters, integrator properties, solver tolerances, initial conditions, etc …

In a first approach, dynamic simulations are done under MatLab numerical interface environment. After obtaining the desired dynamic behaviour for the system and the most suitable integrator, equations of motion can be easily traduced to faster computer languages such as C++.

11 NUMERICAL EXPERIMENTS

The whole model has been introduced until now, but it is necessary to test it in order to know that it works properly. In this section, some manoeuvres are developed and tested in order to validate the proper behaviour of the model. All the tests are prepared so that a minimum number of parameters are involved in each one. This way, the manoeuvres depend on only few of the parameters and some of them can be optimised. In this section only the forklift truck model behaviour results are going to be shown. Integration time results are going to be exposed in the next section.

Some of the manoeuvres are going to be introduced together since they differ only on the value of one of the parameters. The first group of manoeuvres are quite simple while the last group involve accelerating, rear wheel turning and load movement. The number identifiers for the manoeuvres are {1,2,3,4,5,10,11,12,14,15,17,18} because they belong to a larger number of manoeuvres.

11.1 Manoeuvres 1 to 3 (starting manoeuvres)

Since these three manoeuvres are almost the same, they are explained in the same sub-section. The simulation starts in a state in which the forklift truck is stopped. A different torque is exerted in the wheels in each of the simulations. In the first two manoeuvres a constant torque (300 N·m for the first and 600 N·m for the second) is exerted from the beginning of the simulation until the end of it. But in the third one, a linearly progressive torque is exerted. These three simulations try to show the behaviour of the model when accelerating. Moreover, the k parameter of the F_X forces can be optimised with these simulations. The results for the exerted torque, slip ratio, F_X forces and pitch angle are shown in the next figure (9):

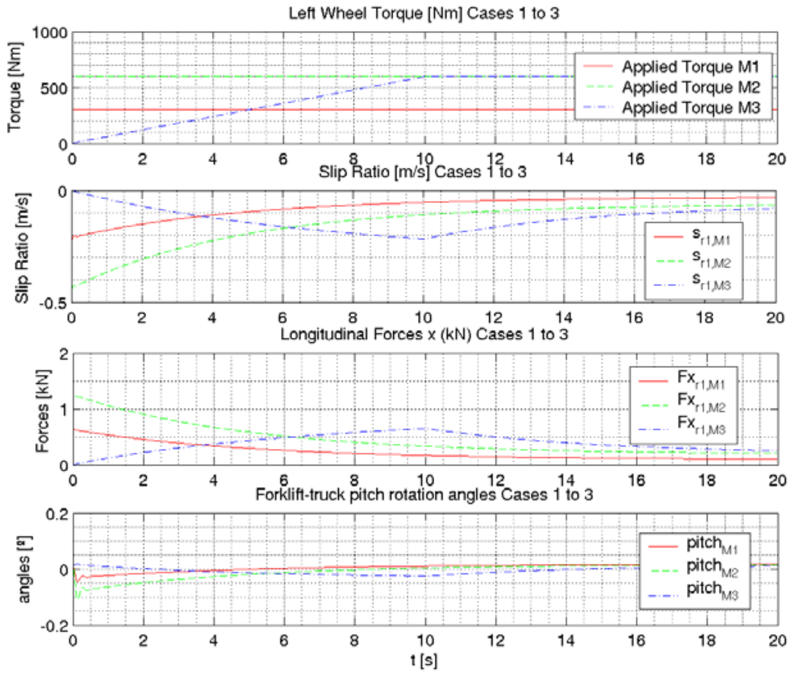


Figure (9)

11.2 Manoeuvres 4 and 5 (braking manoeuvres)

In these simulations two model characteristics are tested: the braking torque changing functions and the ability of the model to overturn. Both of the manoeuvres are performed with a load of 500 kg situated at 2 metres height. They both start with a progressive torque exerted in the first 10 seconds of simulation. But the braking torque is different for each of them: 1200 N·m for the first and 500 N·m for the second manoeuvre. This way, the load decelerated generates a momentum that makes the forklift truck tend to overturn forward.

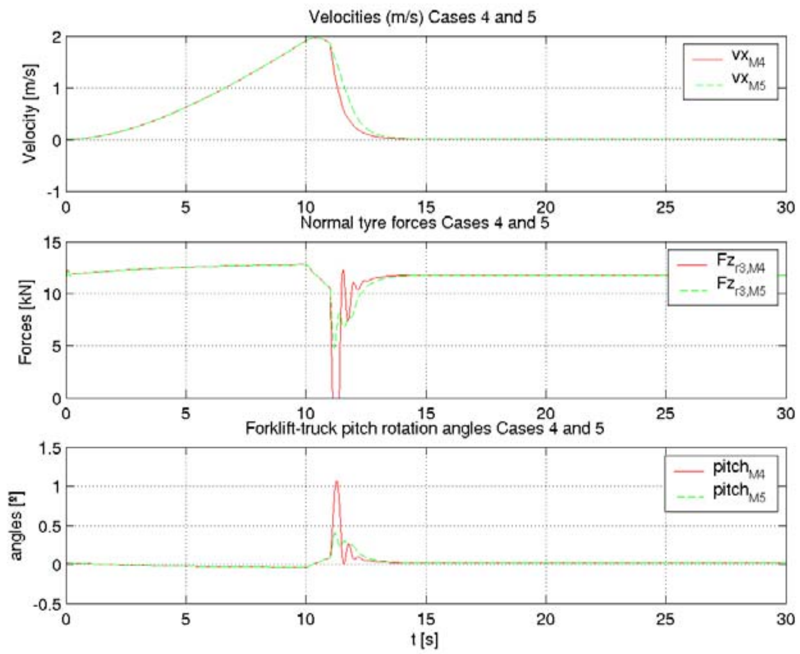


Figure (10)

In the fourth manoeuvre the rear wheel looses contact with the ground and the time spent for braking is 1.5 seconds. The pitch angle reaches a maximum value of 1º. This justifies the small rotation model to be used. In the 5th manoeuvre, the wheel nearly looses contact and spends 2 seconds for braking.

11.3 Manoeuvres 10 to 12 (change of direction manoeuvres)

In these simulations a change of direction manoeuvre is tested. The rear wheel driven angle changes in a continuous and differentiable manner so that no force are exerted sharply. The three rear wheel angle functions with respect to time are all the same but the value of the maximum angle parameter. The first of all reaches $\alpha = \pi / 8$, the second one $\alpha = \pi / 4$, while the last one reaches up to $\alpha = \pi / 3$. All of them spend 5 seconds for changing the rear wheel angle from zero to the maximum and back to zero. It can be seen in the figure (11) how the vertical forces increase in the outside when turning.

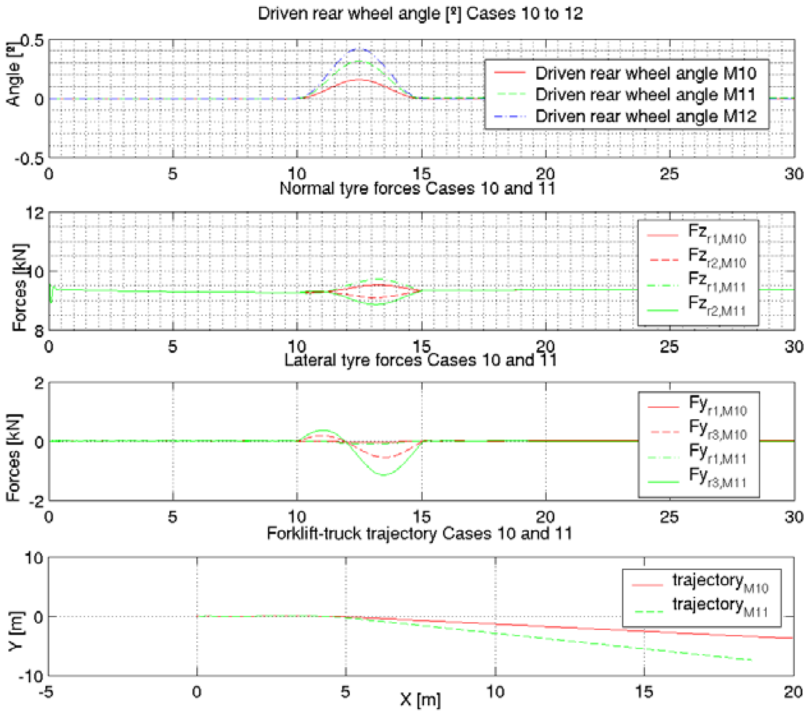


Figure (11)

11.4 Manoeuvres 14 and 15 (turning with unbalanced load)

In these two simulations, the effect of carrying an unbalanced heavy load is tested. Moreover, the effect of turning to the most unfavourable side with such unbalanced load is also tested. The load weighs 1500 kg and is moved 0.75 m to the left hand side of the forklift truck. The load is 2 m height.

Up to the first 15 seconds of simulation the conditions and the behaviour of the two simulations are the same. It can be seen that the forklift truck turns right although the rear wheel remains with a zero angle rotation. This happens because since the left wheel suffers more vertical load (F_Z), the torque that our torque distributing control system exerts to the left wheel is bigger for that wheel and thus the forklift truck turns right.

From t=15s to t=20s, the rear wheel turns up to a maximum rotation angle and turns back to zero. This maximum angle equals $\pi / 6$ in the first simulation and $\pi / 3$ in the second one.

It can be seen that in the 15th manoeuvre the two forward wheel's slip angles exceed the α_0 limit value and, therefore, the $F_{y\_r1}^{M15}$ and $F_{y\_r2}^{M15}$ forces reach saturation point. Moreover, it can be deduced from the F_Z figures that the right wheel almost looses contact with ground in the 15th manoeuvre.

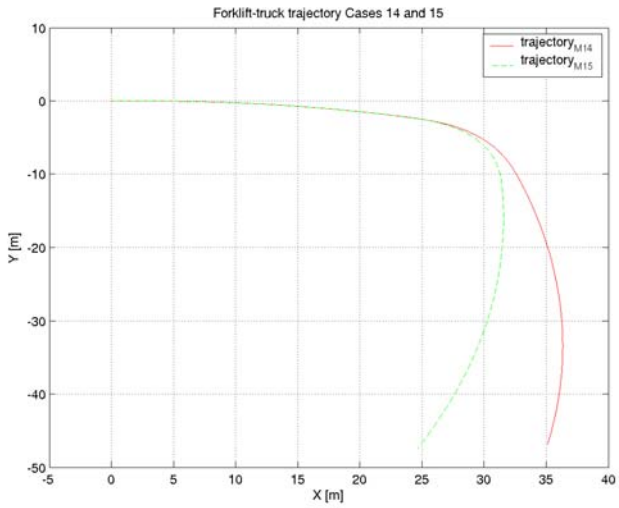
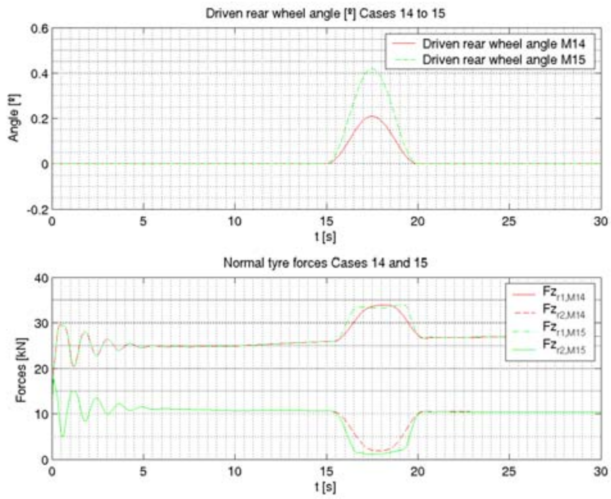
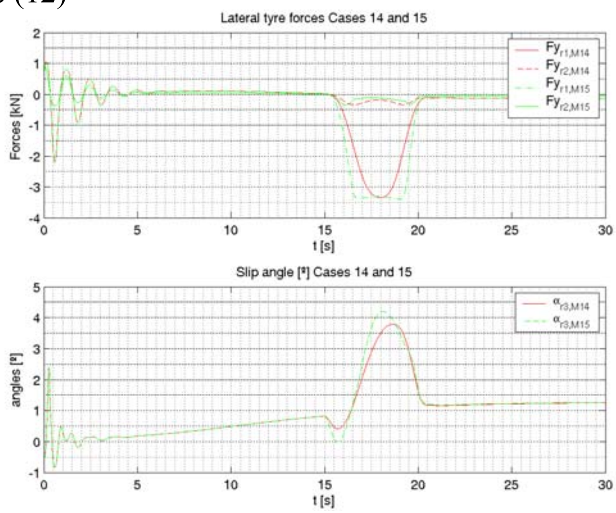


Figure (12)



Figures (13-14)



Figures (15-16)

11.5 Manoeuvres 17 and 18 (zig-zag-like trajectory)

In these two simulations a zig-zag-like trajectory is going to be tested. Trying to generate such a trajectory a rear wheel time dependant function has been prepared which it varies in a sinusoidal manner. The frequency of the sinusoidal is the same in both simulations (0.2 Hz) but the amplitude of the angle equals $\pi / 4$ in the first and $\pi / 6$ in the second one.

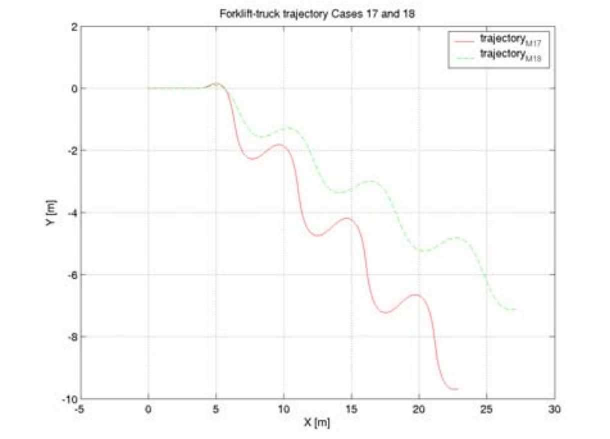
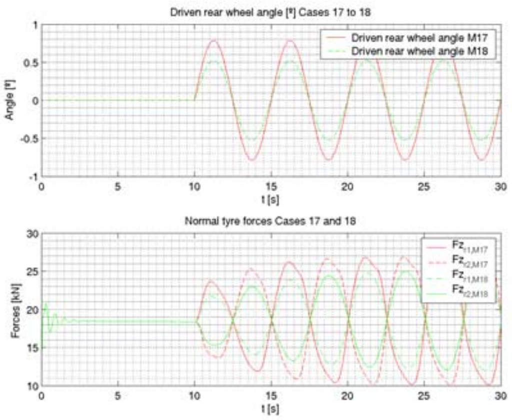
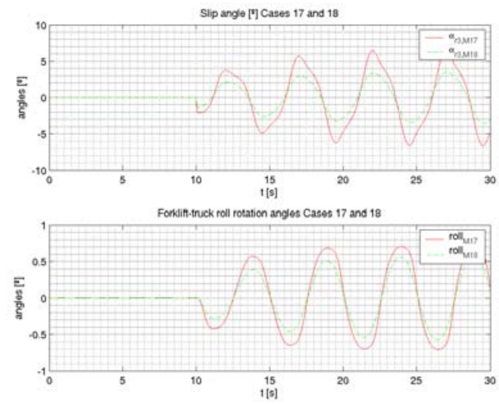


Figure (17)



Figures (18-19)



Figures (20-21)

12 INTEGRATION TIME RESULTS

From real-time dynamics point of view, stiffness of the mechanical system and continuity of the input (forcing) functions are two important things that one should always bear in mind: stiffness coefficients of constitutive elements must be carefully chosen in order to obtain an acceptable response without severe time-step requirements.

Input functions, taken from peripheral every *n* integrator steps, are discrete values that must be smoothed in order to avoid force discontinuities. Moreover, input functions can be either driven coordinates itself (DC) or driven torques as a function of the input signal of the peripherals.

In this comparative, the behaviour of the system has been tested for a critical manoeuvre (case 4) and for a normal operation manoeuvre (case 17) with three different integrators:

- Explicit Runge Kutta 4-order fixed step size integrator (rk4)
- Explicit Runge-Kutta (4,5)-order, Dormand-Prince pair, integrator (Ode45)
- Modified Rosenbrock 2-order variable step size integrator (Ode23s)

Figures (22) and (23) refer the solution for case 4 manoeuvre when using each of the above integrators. As shown in figure (22), the stiffness of the problem makes ode45 integrator step-size to be unacceptable for real-time dynamics. Moreover, as can be seen in figure (23), rk4 calculated solution might present substantial errors in comparison with the correct one.

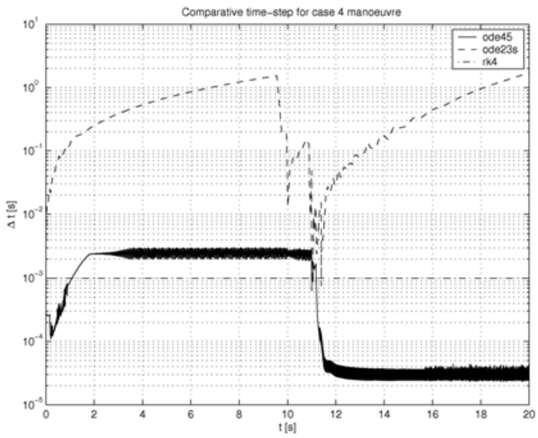


Figure (22)

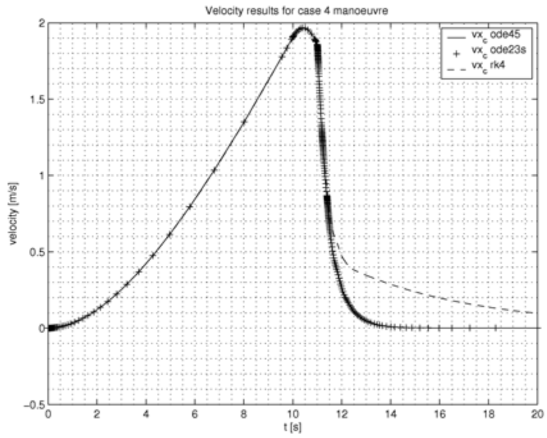


Figure (23)

Figures (24) and (25) refer the solution for case 17 manoeuvre. As can be seen in figure (24), step-size of ode45 solution turns out to be acceptable in this case. Moreover, all three integrators give analogous results for this normal operating manoeuvre.

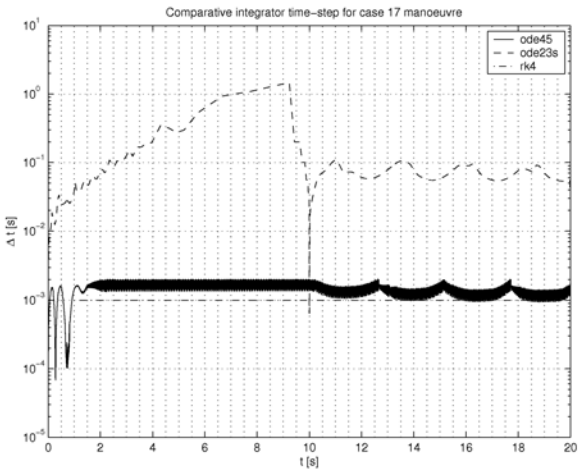


Figure (24)

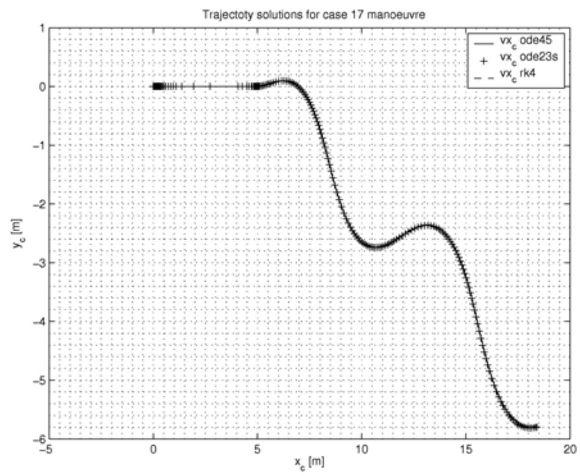


Figure (25)

The conclusion of the simple experiment carried out in this section (see table 1) is that the stiffness of the differential problem may change depending on the characteristics of the forcing functions. Consequently, smoothing functions and maximum torque and (DC) variations must be carefully analysed if an explicit integrator (rk4, ode45) is used. Otherwise, an implicit integrator will be required.

	Case 4	**Case 17**
Ode45	287004	15289
Ode23s	245	261
Rk4	20000	20000

Table 1: Time-steps for a 20 s. simulation

13 CONCLUSIONS

- A forklift truck dynamic model has been developed in order to make use of it in a real-time man-in-the-loop simulator for preventing the industrial accidents derived from the use of forklift trucks. The number of generalised coordinates has been optimised by using relative coordinates, in order to reach real-time simulation requirements.
- Some suitable input forcing functions have been developed (ground to wheel forces) in order not to drive the model in a kinematical way. Moreover, a torque distribution function has been developed to emulate the control system of the real electric fortlift trucks. This function has demonstrated to be suitable in both open and closed curves.
- The stiffness of the differential problem may change depending on the characteristics of the forcing functions. This way, although some manoeuvres could be integrated with explicit schems, the implicit algorithms will have to be used in order to achive stability and reduce computational cost in some sharp manoeuvres.

REFERENCES

[1] J. García de Jalón. *Kinematic and Dynamic simulation of Multibody Systems. The real-time chalenge. Springer-Verlag 1993*

[2] Hans B. Pacejka *Tyre and vehicle dynamics*. Elsevier, 2002.

[3] Mike Blundell and Damian Harty. The Multibody System Approach to Vehicle Dynamics. Elsevier, 2004.

[4] J. Ros et al. *3D_MEC: Un programa para el análisis numérico, simbólico y gráfico en mecánica vectorial y analítica (presentación)*, XIII Congreso Nacional de Ingeniería Mecánica (1998).

[5] J. Gil et al. *Planteamiento de ecuaciones de sistemas multicuerpo rígidos con independencia del tipo de coordenadas utilizando una librería basada en MatLab-Maple*. Congreso de Métodos Numéricos en Ingeniería Mecánica, Lisboa 2004.

[6] J. Gil. *Preprocesador para la Simulación Dinámica de Sistemas Multi-Cuerpo basado en Álgebra Simbólica*. Tesis Doctoral. Universidad Pública de Navarra, 2005. http://www.imac.unavarra.es/jgil.