Modelling, Simulation and Control of the Walking of Biped Robotic Devices—Part I : Modelling and Simulation Using Autolev

*inventions*

MDPI

*Article*

# Modelling, Simulation and Control of the Walking of Biped Robotic Devices—Part I : Modelling and Simulation Using Autolev

**Giuseppe Menga *,† and Marco Ghirardi †**

Control and Computer Engineering Department, Politecnico di Torino, Corso Duca Degli Abruzzi 24, 10124 Torino, Italy; marco.ghirardi@polito.it

* Correspondence: menga@polito.it; Tel. +39-011-0907272

† These authors contributed equally to this work.

Academic Editor: Chien-Hung Liu

Received: 23 November 2015; Accepted: 7 March 2016; Published: 22 March 2016

**Abstract:** A biped robot is a mechanical multichain system. The peculiar features, that distinguishes this kind of robot with respect to others, e.g., industrial robots, is its switching nature between different phases, each one is the same mechanics subject to a different constraint. Moreover, because these (unilateral) constraints, represented by the contact between the foot/feet and the ground, play a fundamental role for maintaining the postural equilibrium during the gait, forces and torques returned must be continuously monitored, as they pose stringent conditions to the trajectories that the joints of the robot can safely follow. The advantages of using the Kane's method to approach the dynamical model (models) of the system are outlined. This paper, divided in three parts, deals with a generical biped device, which can be an exoskeleton for rehabilitation or an indipendent robot. Part I is devoted to modelling and simulation, part II approaches the control of walk in a rectilinear trajectory, part III extends the results on turning while walking. In particular, this part I describes the model of the biped robot and the practicalities of building a computer simulator, leveraging on the facilities offered by the symbolic computational environment Autolev that complements the Kane's method.

**Keywords:** mechanical multi-chain; non-holonomic systems; hybrid complementarity systems; biped robotics; simulation; Kane's method; object oriented design

## 1. Introduction

A biped robot is a particular mechanical multichain system, which, as the time evolves, switches between phases (single stance, double stance, each one divided by subphases), characterized by different constraints. This has several consequences:

**From a postural equilibrium point of view** because these (unilateral) constraints, represented by the contact between the foot/feet and the ground, play a fundamental role for maintaining the postural equilibrium during the gait, their returned forces and torques pose stringent conditions to the trajectories that the joints of the robot can safely follow, and must be continuously monitored;

**From the robot control point of view** the changing constraints modify the number of degrees of freedom (DOF) of the dynamics, that can result underactuaded or overactuaded depending on the ground contacts, and the controller itself must be a switching system.

The authors have been involved for several years in modelling and control of biped robots for entertainment [1] and exoskeletons for postural rehabilitation [2], so they were faced from the

beginning with the problem to provide the different projects with a reliable simulator. Commercial products such as ADAMS [3] were considered too limited from several points of view:

1. the need to build an hybrid system switching among several phases;
2. obtaining from the non-linear dynamical equations matrices of the linearized model to be used for designing a (linear) control;
3. deriving explicitly computer code for the expressions of Jacobians matrices, kinematics, and in general of mechanical quantities needed for analysis and then to be embedded in the control;
4. obtaining from the ground constraint the theoretical expressions of reaction forces/torques on the feet needed to evaluate the postural equilibrium, to balance the weight and offer compliance to the swing foot in the contact with the ground.

For these reasons the approach to build a completely open simulator, with full control of the programming was pursued.

Hybrid complementarity dynamical systems [4,5] model the behaviour of biped robots during walk. They are the basis of the control of bipeds through forward dynamics (see [6] section 16.3). These models correctly treat the contact between the ground and the feet as unilateral constraints. We follow here a different approach by considering the contact as a bilateral constraint and describing the system as different non-holonomic dynamical models, one for each phase of the gait. This has, with respect to classical approaches, some advantages: the models are simpler, as they don't require the iterative computation of lagrangian multipliers needed to solve the constraints, they explicitly show for each phase the degrees of freedom of the dynamics, allowing to choose independent and depend control variables, and offering analitically reaction forces and torques at the ground. However, this approximation has one drawback: these models can handle patological walking situations of slipping or detachment of the foot from the ground only indirectly. To take full advantage of the approach, instead of the standard equations proposed in the literature the Kane's method [7] has been adopted in order to derive the multi-chain dynamics. This method was originated in the spacecraft dynamics community, and apparently is not widely known in the robotic, and particularly biped robotic field, yet. A primer of the method will be given in one of the next sections, but just as introduction one of its advantages, with respect to other methods, is the capability to derive unitarily from the biped free in the space the several lower order non-holonomic representations in the presence of different constraints. For a more detailed discussion on simulators of multibody chains, specifically using the Kane's method see [8]. Moreover the Kane's method is complemented with a symbolic computational environment, originally called Autolev and now MotionGenesis Kane 5.x [9], that generates all mathematical expressions needed for modelling and design controls, and produces prototypical source code in Fortran, Matlab or C of the equations in ordinary differential form (ODEs) even for non holonomical constrained systems, which can be simulated using standard solvers.

In this framework are easily available, also, expressions of a variety of mechanical quantities needed for the analysis, such as linearization matrices, kinematics, Jacobians, explicit reaction forces/torques from the constraints, kinetic energy, generalized impulses, generalized momentum, *etc*.

The fragments of C software generated by Autolev have been casted, with few small modifications, into a framework of Object Oriented Design in C++ obtaining the hybrid nonlinear system simulation we needed and an easy interface with control modules and reference trajectory generators. Moreover, some of the mathematical expressions present in the model can be extracted from the simulator and in the future embedded in the real time hardware to become integral part of the control.

This paper extends the study of a lower limb exoskeleton for postural rehabilitation [2] and is composed by three parts. This first part is devoted to modelling and simulation, part II approaches the control of walk in a rectilinear trajectory, part III extends the results on turning while walking.

The organization of Part I is as follows: a review of previous work, specifically hybrid complementary dynamical systems, is presented in Section 2; in Section 3 a biped robot and its mechanical model is described; in Section 4 the general scheme of the simulator is presented.

## 2. Hybrid Complementarity Dynamical Systems

A small review of these models is taken from the related literature [10–13].

A biped robot is a three-dimension rigid multichain. Basically, the internal configuration variables $\mathbf{q}_1$ are the ones of the joint space $Q$. However, it is usual to consider a walking robot as a system free in the space, but subject to non-constant unilateral constraints. Hence, the configuration space is $Q \oplus \mathbb{R}^6$, where the six dimensional displacement of a given body is parameterized by $\mathbf{q}_2$. Assuming that the body structure is rigid, continuous dynamics can be expressed under the following Lagrangian form

$$\mathbf{M}(\mathbf{q}) \cdot \ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \cdot \dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}) + \mathbf{\Gamma}_{ext}, \tag{1}$$

where $\mathbf{q} = (\mathbf{q}_1, \mathbf{q}_2)^T \in \mathbb{R}^n$ is the parametrization vector of the whole configuration space of the biped considered as free in 3D space, $\mathbf{\Gamma} \in \mathbb{R}^n$ is the generalized efforts vector (function of the joint actuation torque $\boldsymbol{\tau} \in \mathbb{R}^m$), $\mathbf{M}$ is the inertia matrix, $\mathbf{C}$ are the centrifugal, gyroscopic, and Coriolis contribution, and $\mathbf{G}$ is the generalized gravity force vector. Note that the set $(\mathbf{q}, \dot{\mathbf{q}})$ constitutes the state of the biped, in the sense of the theory of dynamic systems. $\mathbf{\Gamma}_{ext}$ are torques generated by external reactions (ground contacts, interaction with objects) and they can be expressed as follows:

$$\mathbf{\Gamma}_{ext} = \mathbf{J}(\mathbf{q})^T \cdot \boldsymbol{\lambda}(\mathbf{q}, \dot{\mathbf{q}}) \tag{2}$$

where $\mathbf{J}(\mathbf{q})$ is the Jacobian matrix of the points of the biped to which the external forces are applied and $\boldsymbol{\lambda}(\mathbf{q}, \dot{\mathbf{q}})$ are the Lagrangian multipliers corresponding to the amplitudes of these reactions.

The vector $\boldsymbol{\lambda}$ can be split in:

$$\boldsymbol{\lambda} = (\boldsymbol{\lambda}_n, \boldsymbol{\lambda}_t)^T \tag{3}$$

where $\boldsymbol{\lambda}_n$ and $\boldsymbol{\lambda}_t$ are the normal and tangential components of the reactions, respectively. $\boldsymbol{\lambda}_n$ satisfies the complementarity condition

$$\boldsymbol{\lambda}_n^T \cdot f(\mathbf{q}) = 0, \boldsymbol{\lambda}_n \geq 0, f(\mathbf{q}) \geq 0 \tag{4}$$

with $\mathbf{J}(\mathbf{q}) = \{\partial f_i(\mathbf{q})/\partial q_j\}, i, j = 1 \cdots n$ and $\boldsymbol{\lambda}_t$ the Coulomb's friction condition for each component:

$$|\lambda_{t_i}| \leq \mu \lambda_{n_i} \tag{5}$$

Moreover, switching from single to double stance, the impact and restitution law has to be considered in setting the initial condition for the next phase.

Note, from an implementation point of view, that during simulation $\boldsymbol{\lambda}$ has to be numerically evaluated iteratively.

## 3. A Biped Robot

### *3.1. Mechanical Modelling*

As it is known, the model of a multi-body system like a biped robot can be derived using different, essentially equivalent, methods. However, the ease of use of the various methods differs; some are more suited for multibody dynamics than others. The Newton-Euler method is comprehensive in that a complete solution for all the forces and dynamic variables are obtained, but it is inefficient. Applying the Newton-Euler method requires that force and moment balances be applied for each body taking in consideration every interactive and constraint force. Therefore, the method

is inefficient when only a few of the system's forces need to be solved for. Lagrange's Equations provides a method for disregarding all interactive and constraint forces that do not perform work. The major disadvantage of this method is the need to differentiate scalar energy functions (kinetic and potential energy). This is not much of a problem for small multibody systems, but becomes an efficiency problem for large multibody systems.

In this work, instead, the so-called Kane's method ([7]) has been chosen. This method is particularly interesting in this case because it is equally applicable to either holonomic or non-holonomic systems and, for non-holonomic systems, without the need to introduce Lagrangian multipliers. Briefly, the main contribution of the Kane's method is that, through the concepts of motion variables (later called generalized speeds), the vectors of partial velocities and partial angular velocities, generalized active forces and generalized inertia forces, the dynamical equations are automatically determined, enabling forces and torques having no influence on the dynamics to be eliminated early in the analysis. Early elimination of these noncontributing forces and torques greatly simplifies the mathematics and enables problems having greater complexity to be handled. More into details:

**Generalized coordinates and speeds** A multi-body system, which possesses $n$ degrees of freedom, is represented by a state with a $n$ dimensional vector $\mathbf{q}$ of configuration variables (*generalized coordinates*) and an identical dimension vector $\mathbf{u}$ of *generalized speeds* called also *motion variables*, that could be any nonsingular combination of the time derivatives of the generalized coordinates that describe the configuration of a system. These are the kinematical differential equations:

$$u_r = \sum_{i=1,\cdots,n} Y_{ri}\dot{q}_i, r = 1, \cdots, n \tag{6}$$

$Y_{ri}$ may be in general nonlinear in the configuration variables so that the equations of motion can take on a particularly compact (and thus computationally efficient) form with the effective use of generalized speeds.

**Partial velocities and angular velocities** Partial velocities of each point (partial angular velocity of each body) are the $n$ three-dimensional vectors expressing the velocities of that point (angular velocity of that body) as a linear combination of the generalized speeds. Let be $\mathbf{v}^P$ the translational velocity of a point $P$ and $\boldsymbol{\omega}^B$ the rotational velocity of a body $B$ with respect to the inertial reference frame, then

$$\begin{array}{l} \mathbf{v}^P = \sum_{r=1..n} \mathbf{v}_r^P u_r \\ \boldsymbol{\omega}^B = \sum_{r=1..n} \boldsymbol{\omega}_r^B u_r \end{array} \tag{7}$$

where $\mathbf{v}_r^P$ and $\boldsymbol{\omega}_r^B$ are the $r_{th}$ partial velocity and partial angular velocity of $P$ and $B$, respectively.

**Generalized active and inertia forces** The $n$ generalized forces acting on a system are constructed by the scalar product (projection) of all contributing forces and torques on the partial velocities and partial angular velocities of the points and bodies they are applied to.

Let us consider a system composed by $N$ bodies $B_i$, where the torque $\mathbf{T}^{B_i}$, and force $\mathbf{R}^{P_i}$ applied to a point $P_i$ of $B_i$ are the equivalent resultant (*"replacement"* [7]) of all active forces and torques applied to $B_i$. Then

$$F_r^{B_i} = \boldsymbol{\omega}_r^{B_i} \cdot \mathbf{T}_i^{B_i} + \mathbf{v}_r^{P_i} \cdot \mathbf{R}_r^{P_i} \tag{8}$$

is the $r_{th}$ generalized active force acting on $B_i$ and

$$F_r = \sum_{i=1,\cdots,N} F_r^{B_i} \tag{9}$$

the $r_{th}$ generalized active force acting on the whole system. Identically for the inertia forces, indicated as $F_r^*$.

The dynamical equations for an $n$ degree of freedom system are formed out from generalized active and inertial forces $F_r^*$

$$F_r + F_r^* = 0, r = 1, \cdots, n. \tag{10}$$

These are known as Kane's dynamical equations.

They result in a $n$ dimensional system of second order differential equations ($2n$ order state variable representation) on generalized coordinates and speeds

$$\bar{\mathbf{M}}(\mathbf{q})\dot{\mathbf{u}} + \bar{\mathbf{C}}(\mathbf{q}, \mathbf{u})\mathbf{u} + \bar{\mathbf{G}}(\mathbf{q}) - \bar{\mathbf{\Gamma}}(\mathbf{q}, \mathbf{u}, \boldsymbol{\tau}) = \mathbf{0}, \tag{11}$$

where the parameter definitions are similar but not identical of Equation (1) and more efficient computationally [8].

**Non-holonomic constraints** When $m$ constraints on the motion variables are added to the model, only $n - m$ generalized speeds are independents.The system is, then, called a non-holonomic system. The *non-holonomic* constraints are expressed as a set of $m$ linear relationships between dependent and independent generalized speeds of the type

$$u_r = \sum_{i=1,\cdots,p} A_{ri} u_i, r = p + 1, \cdots, n, \tag{12}$$

with $p = n - m$. In this case, selected the independent speeds, the Kane's method immediately offers the minimal $2p$ order state variable representation from

$$\widetilde{F}_r + \widetilde{F}_r^* = 0, r = 1, \cdots, p, \tag{13}$$

where Kane calls $\widetilde{F}_r$ and $\widetilde{F}_r^*$ non-holonomic generalized active and inertial forces, while the remaining $m$ original redundant equations resolve themselves in the expressions of the $m$ reaction forces/torques returned by the constraints (because the Kane's method is fundamentally based on the projection of forces on a tangent space on which the system dynamics are constrained to evolve, spanned by the partial velocities, reaction forces/torques result from the projection on its null-space).

Obviously, the choice of independent speeds is not unique, and they are usually found by inspection. However, a systematic way may be adopted considering that Equation (12) in the implicit vectorial form defines the null-space of $\boldsymbol{u}$. Split $\boldsymbol{u}$ in independent and dependent generalized speeds $\boldsymbol{u}_1, \boldsymbol{u}_2$ and rewrite Equation (12), introducing the matrix $\mathbf{A}$, $pxm$, as

$$(\mathbf{A}^T, -\mathbf{I}_{mxm}) \cdot \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix} = 0. \tag{14}$$

The columns of

$$\begin{bmatrix} \mathbf{A} \\ -\mathbf{I}_{mxm} \end{bmatrix} \tag{15}$$

and

$$\begin{bmatrix} \mathbf{I}_{pxp} \\ \mathbf{A}^T \end{bmatrix} \tag{16}$$

span, respectively null-space and range of $\mathbf{u}$. In order to obtain the most robust variables, permute the components of $\mathbf{u}$ in the independent generalized speeds so as to minimize the conditioning number of either equation (15) or (16).

Moreover, it is always possible to handle an holonomic (configuration) constraint as if it is non-holonomic, that is, to treat it as a motion constraint. This is particularly advantageous in biped

robotics, in fact, considering foot/feet contact to the ground as non-holonomic contraints, from the reaction forces/torques returned from the constraint, the Zero Moment Point ($ZMP$) coordinates, and hence the Center of Pressure ($CoP$), can readily found as shown in the following subsection.

### *3.2. Robot Configuration*

In this section the specific biped robotic structure we exploit in our practical applications in described. The key bodies and points are depicted in the Figures 1 and 2 where the frontal and the sagittal view of the robot are depicted. It is composed of 7 bodies: feet, legs, thighs, hat (HAT: head, arm and trunk), connected by 12 joints: 2 ankles, 2 knees and 2 hips. Each ankle has 3 DOF, the knee 1 and the hip 2. In order to guarantee a natural gait the feet are allowed to rotate along $y$. This is equivalent to a hinge on the toes or on the heels of the foot/feet on the ground, resulting in 13 or 14 joints that can be potentially controlled, in single or double stance, respectively. In fact, to guarantee a human like walk it is mandatory that toe alone, or toe and heel joints are active [14–16].

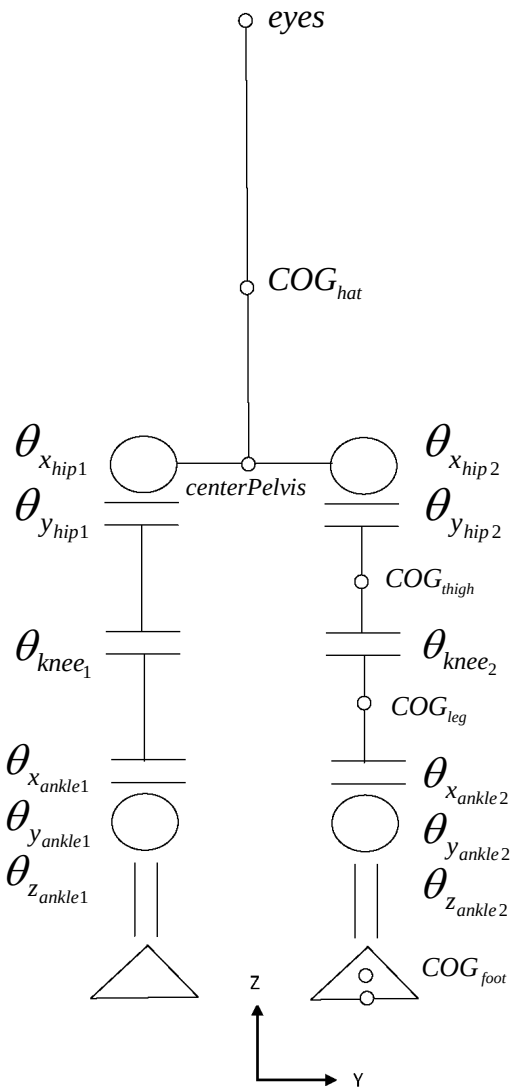


**Figure 1.** The robot: frontal plane - joint configuration.

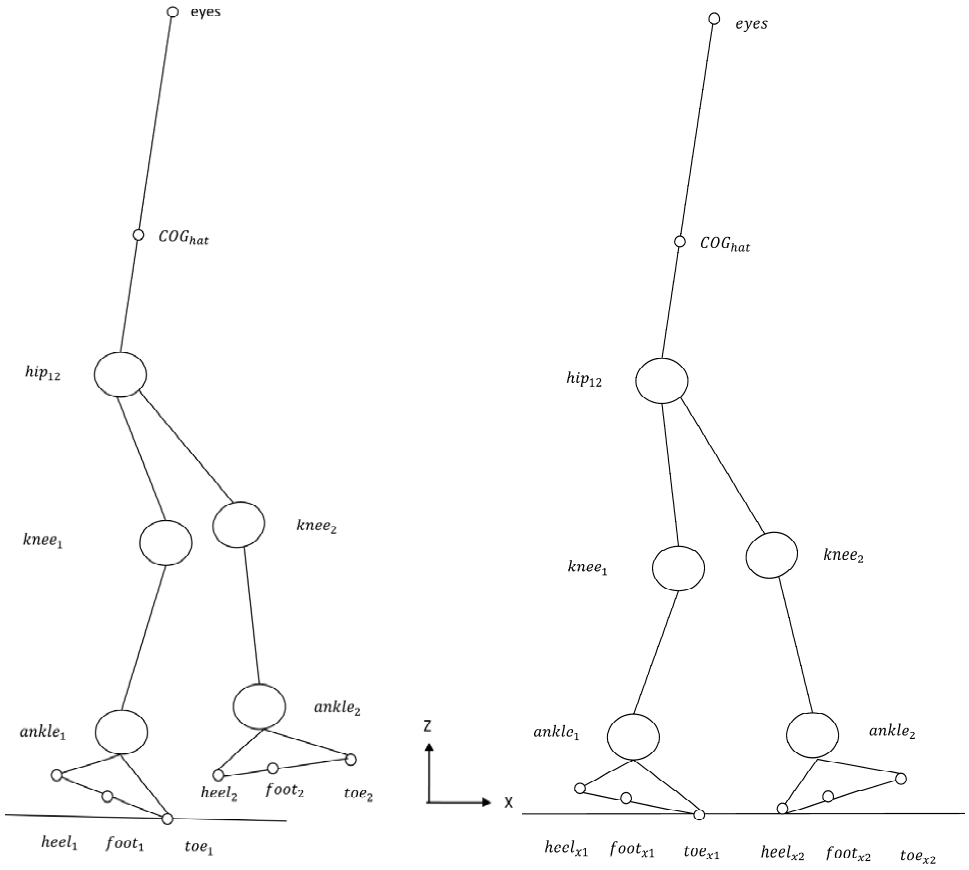


**Figure 2.** The robot: sagittal plane—single and double stance.

Assuming the system free to move in the sagittal and frontal planes, but for simplicity not to rotate along the $z$ axis (the walk is considered to be along a straight line without turns), it is characterized by 17 *DOF* and an identical number of generalized variables and speeds, chosen here as the derivatives of the generalized variables. These are (angles are characterized by the rotation axis, the joint name and the limb index; positions by the axis, the point name and the limb index):

- The three positions on space of a reference point of $foot_1$, were reaction forces are applied, taken at the center of the toe: $x_{toe_1}, y_{toe_1}, z_{toe_1}$.
- The two angles of the frame of foot 1: $\theta_{x_{foot_1}}, \theta_{y\,foot_1}$.
- The joint angles (as depicted in Figure 1): $\theta_{x_{ankle_1}}$, $\theta_{y_{ankle_1}}$,$\theta_{z_{ankle_1}}$, $\theta_{knee_1}$, $\theta_{x_{hip_1}}$, $\theta_{y_{hip_1}}$, $\theta_{x_{hip_2}}$, $\theta_{y_{hip_2}}$, $\theta_{knee_2}$, $\theta_{x_{ankle_2}}$, $\theta_{y_{ankle_2}}$ and $\theta_{z_{ankle_2}}$.

Indicate with pedix 1 the right leg and pedix 2 the left leg, and use conventionally foot 1 as the supporting foot. The two configurations of single and double stance are defined as follows:

***Single Stance***: The exoskeleton is sustained by $foot_1$. Both feet are allowed to rotate around $y$ only, in particular $foot_1$ is connected to the ground by a hinge at the $toe_1$ point, and it is imposed that the trunk is oriented straight ahead. Then, the three translational velocities of $toe_1$ $\dot{x}_{toe_1}, \dot{y}_{toe_1}, \dot{z}_{toe_1}$, the three rotational velocities $\dot{\theta}_{x_{foot_1}}, \dot{\theta}_{x_{foot_2}}, \dot{\theta}_{z_{foot_2}}$ of both feet and the rotational velocity of the trunk $\dot{\theta}_{z_{hat}}$ are zero (motion constraints). In these conditions the system has 10 *DOF* (7 have been constrained out of the original 17). The unique possible choices of independent generalized speeds is:

$$\begin{aligned}\dot{\boldsymbol{\theta}}_s = [&\dot{\theta}_{y_{foot_1}}, \dot{\theta}_{x_{ankle_1}}, \dot{\theta}_{y_{ankle_1}}, \dot{\theta}_{knee_1}, \dot{\theta}_{x_{hip_1}}, \dot{\theta}_{y_{hip_1}},\\ &\dot{\theta}_{x_{hip_2}}, \dot{\theta}_{y_{hip_2}}, \dot{\theta}_{knee_2}, \dot{\theta}_{y_{ankle_2}}]^T\end{aligned} \tag{17}$$

controlled by 10 torques at the corresponding joints

$$\begin{aligned}\boldsymbol{T}_s = [&T_{y_{foot_1}}, T_{x_{ankle_1}}, T_{y_{ankle_1}}, T_{knee_1}, T_{x_{hip_1}}, T_{y_{hip_1}},\\ &T_{x_{hip_2}}, T_{y_{hip_2}}, T_{knee_2}, T_{y_{ankle_2}}]^T.\end{aligned} \tag{18}$$

Note that, if the toe of $foot_1$ is passive we have an under-actuated system.

On the other hand, the motion constraints resolve themselves in 3 forces: $F_{x_{foot_1}}$, $F_{y_{foot_1}}$, $F_{z_{foot_1}}$, and 4 torques: $T_{x_{foot_1}}$, $T_{z_{ankle_1}}$, $T_{x_{ankle_2}}$, $T_{z_{ankle_2}}$ of reaction. The first four, along with the control torque $T_{y_{foot_1}}$, characterizing the foot reactions from the ground

$$\boldsymbol{F}_{foot_1} = [F_{x_{foot_1}}, F_{y_{foot_1}}, F_{z_{foot_1}}, T_{x_{foot_1}}, T_{y_{foot_1}}]^T \tag{19}$$

are very important in determining the $ZMP$, the remaining three, vice versa, not essential for posture, will be ignored (The corresponding joints will be separately controlled with respect to the main multivariable control, in order to keep the biped trunk, as well as the swing foot, oriented straight ahead).

***Double stance***: when the exoskeleton is in double stance, both feet are on the ground. As they can rotate around y, it is assumed that $foot_1$ is connected to the ground by a hinge, as before, at $toe_1$ and the $foot_2$ at $heel_2$, and the center of the heel $(x_{heel_2}, y_{heel_2}, z_{heel_2})$ is the reference point where reaction forces are applied. Hence, the system loses 3 more degrees of freedom resulting in 7. There are four possible choices of independent generalized speeds, however, the best conditioned, according to Section 3.1, is:

$$\begin{aligned}\dot{\boldsymbol{\theta}}_d = [&\dot{\theta}_{y_{foot_1}}, \dot{\theta}_{x_{ankle_1}}, \dot{\theta}_{y_{ankle_1}}, \dot{\theta}_{knee_1}, \dot{\theta}_{y_{hip_1}},\\ &\dot{\theta}_{knee_2}, \dot{\theta}_{y_{ankle_2}}]^T\end{aligned} \tag{20}$$

and the system in this condition is controlled by the 7 torques at the corresponding joints

$$\begin{aligned}\boldsymbol{T}_d = [&T_{y_{foot_1}}, T_{x_{ankle_1}}, T_{y_{ankle_1}}, T_{knee_1}, T_{y_{hip_1}},\\ &T_{knee_2}, T_{y_{ankle_2}}]^T.\end{aligned} \tag{21}$$

Note that the remaining 5 torques, comprehensive of $T_{y_{foot_2}}$ at the hinge of $heel_2$, if it is active,

$$\boldsymbol{T}_{aux} = [T_{x_{hip_1}}, T_{x_{hip_2}}, T_{y_{hip_2}}, T_{x_{ankle_2}}, T_{y_{foot_2}}]^T. \tag{22}$$

are redundant for position control. $T_{x_{ankle_2}}$, which is irrelevant in single stance, along with the whole vector Equation (22), will play a significant role in load balance in double.

The complete list of the motion variables, constraints, reaction forces and torques, joint torques needed for position control, and redundant joint torques for single and double stance are contained in Tables 1 and 2.

Reaction and control force/torques at the foot/feet are particularly important in biped robotics:

$$\boldsymbol{F}_{foot_i} = [F_{x_{foot_i}}, F_{y_{foot_i}}, F_{z_{foot_i}}, T_{x_{foot_i}}, T_{y_{foot_i}}]^T \tag{23}$$

where $i = 1, 2$ and, specifically, the $CoP$ coordinates, derived from the $ZMP$, result in:

$$\begin{aligned}CoP_x &= \frac{-T_{y_{foot_1}} - T_{y_{foot_2}} + x_{toe_1} * F_{z_{foot_1}} + x_{heel_2} * F_{z_{foot_2}}}{F_{z_{foot_1}} + F_{z_{foot_2}}},\\ CoP_y &= \frac{T_{x_{foot_1}} + T_{x_{foot_2}} + y_{toe_1} * F_{z_{foot_1}} + y_{heel_2} * F_{z_{foot_2}}}{F_{z_{foot_1}} + F_{z_{foot_2}}}\end{aligned} \tag{24}$$

where $T_{j_{foot_i}}$, $F_{z_{foot_i}}$, $x_{toe_i}$, $y_{heel_i}$, $i = 1,2$, $j = x,y$ are torques and vertical forces with their points of application on the soles of the feet, $toe_1$ and $heel_2$ . For an in-depth discussion on $CoP$ and $ZMP$ see [6].

**Table 1.** Single Stance.

| Motion Variables | Constraints | Reactions | Position Control | Free Forces/Torques |
|---|---|---|---|---|
| $\dot{\theta}_{y_{foot_1}}$ | | | $T_{y_{foot_1}}$ | $F_{x_{foot_2}}$ |
| $\dot{\theta}_{y_{ankle_1}}$ | | | $T_{y_{ankle_1}}$ | $F_{y_{foot_2}}$ |
| $\dot{\theta}_{knee_1}$ | | | $T_{knee_1}$ | $F_{z_{foot_2}}$ |
| $\dot{\theta}_{y_{hip_1}}$ | | | $T_{y_{hip_1}}$ | $T_{x_{foot_2}}$ |
| $\dot{\theta}_{y_{hip_2}}$ | | | $T_{y_{hip_2}}$ | $T_{y_{foot_2}}$ |
| $\dot{\theta}_{knee_2}$ | | | $T_{knee_2}$ | $T_{z_{foot_2}}$ |
| $\dot{\theta}_{y_{ankle_2}}$ | | | $T_{y_{ankle_2}}$ | |
| $\dot{\theta}_{x_{ankle_1}}$ | | | $T_{x_{ankle_1}}$ | |
| $\dot{\theta}_{x_{hip_1}}$ | | | $T_{x_{hip_1}}$ | |
| $\dot{\theta}_{x_{hip_2}}$ | | | $T_{x_{hip_2}}$ | |
| $\dot{\theta}_{x_{ankle_2}}$ | $\dot{\theta}_{xfoot_2}$ | $T_{x_{ankle_2}}$ | | |
| $\dot{\theta}_{z_{ankle_1}}$ | $\dot{\theta}_{z_{hat}}$ | $T_{z_{ankle_1}}$ | | |
| $\dot{\theta}_{z_{ankle_2}}$ | $\dot{\theta}_{z_{foot_2}}$ | $T_{z_{ankle_2}}$ | | |
| $\dot{\theta}_{x_{foot_1}}$ | $\dot{\theta}_{x_{foot_1}}$ | $T_{x_{foot_1}}$ | | |
| $\dot{x}_{toe_1}$ | $\dot{x}_{toe_1}$ | $F_{x_{foot_1}}$ | | |
| $\dot{y}_{toe_1}$ | $\dot{y}_{toe_1}$ | $F_{y_{foot_1}}$ | | |
| $\dot{z}_{toe_1}$ | $\dot{z}_{toe_1}$ | $F_{z_{foot_1}}$ | | |

**Table 2.** Double Stance.

| Motion Variables | Constraints | Reactions | Position Control | Free Forces/Torques |
|---|---|---|---|---|
| $\dot{\theta}_{y_{foot_1}}$ | | | $T_{y_{foot_1}}$ | |
| $\dot{\theta}_{y_{ankle_1}}$ | | | $T_{y_{ankle_1}}$ | |
| $\dot{\theta}_{knee_1}$ | | | $T_{knee_1}$ | |
| $\dot{\theta}_{y_{hip_1}}$ | | | $T_{y_{hip_1}}$ | |
| $\dot{\theta}_{y_{hip_2}}$ | $\dot{x}_{heel_2}$ | $F_{x_{foot_2}}$ | | $T_{y_{hip_2}}$ |
| $\dot{\theta}_{knee_2}$ | | | $T_{knee_2}$ | |
| $\dot{\theta}_{y_{ankle_2}}$ | | | $T_{y_{ankle_2}}$ | |
| $\dot{\theta}_{x_{ankle_1}}$ | | | $T_{x_{ankle_1}}$ | |
| $\dot{\theta}_{x_{hip_1}}$ | $\dot{z}_{heel_2}$ | $F_{z_{foot_2}}$ | | $T_{x_{hip_1}}$ |
| $\dot{\theta}_{x_{hip_2}}$ | $\dot{y}_{heel_2}$ | $F_{y_{foot_2}}$ | | $T_{x_{hip_2}}$ |
| $\dot{\theta}_{x_{ankle_2}}$ | $\dot{\theta}_{x_{foot_2}}$ | $T_{x_{foot_2}}$ | | $T_{x_{ankle_2}}$ |
| $\dot{\theta}_{z_{ankle_1}}$ | $\dot{z}_{hat}$ | $T_{z_{ankle_1}}$ | | |
| $\dot{\theta}_{z_{ankle_2}}$ | $\dot{\theta}_{z_{foot_2}}$ | $T_{z_{ankle2}}$ | | |
| $\dot{\theta}_{x_{foot_1}}$ | $\dot{\theta}_{x_{foot_1}}$ | $T_{x_{foot_1}}$ | | |
| $\dot{x}_{toe_1}$ | $\dot{x}_{toe_1}$ | $F_{x_{foot_1}}$ | | |
| $\dot{y}_{toe_1}$ | $\dot{y}_{toe_1}$ | $F_{y_{foot_1}}$ | | |
| $\dot{z}_{toe_1}$ | $\dot{z}_{toe_1}$ | $F_{z_{foot_1}}$ | | |
| | | | | $T_{y_{foot_2}}$ |

### *3.3. Unilateral Constraint and Collision*

As a consequence of switching between different non-holonomic models during simulation, unilateral constraints and collisions cannot be ignored.

Clearly, adopting non-holonomic dynamics assuming points of the feet fixed to the ground is valid for bilateral constraints, ignoring eventual detachment from the ground and slipping (Even if possible, it would be too cumbersome to switch among still more different models if each one of the different conditions occur.). In the approaches based on forward dynamics, as it is shown in Section 2, the necessary conditions for satisfying unilateral constraints are directly embedded into the simulation. Vice versa, we adopt here a minimalistic view, noting that in a physiological gait, normally, bilateral constraints on the feet are not assumed to be violated. Hence, we design a priori walking strategies and we test through the simulator that this effectively occurs, by monitoring, a posteriori, reaction forces/torques for the conditions:

$$F_{z_{foot_i}} > 0, i = 1, 2 \tag{25}$$

and

$$|F_{j_{foot_i}}| < \mu F_{z_{foot_i}}, j = x, y, i = 1, 2. \tag{26}$$

Obviously, the control we propose cannot adapt itself to pathological conditions, like a slipping surface.

For the second point, mechanics of the collision of the swing foot to the ground when switching from single to double stance cause the transfer of final conditions of one phase to the initial conditions of the successive, with a reduced number of state variables. With reasonable assumptions of non-slipping and anelastic restitution the reaction impulsive force $\mathbf{F}^{heel_2}$ at the impact and the initial conditions of the generalized speeds for the new phase $\mathbf{u}(t^+)$ can be computed. Also for this aspects we are facilitated by Autolev that offers all the mechanical quantities we need.

The following analysis is based on two concepts: *generalized impulse* and *generalized momentum* [7,17]. Let us indicate, as usual, with $\mathbf{v}_r^{heel_2}$ the r-th component of the partial velocity vectors of $heel_2$ on single stance, the *generalized impulse* on the $heel_2$ at the contact with the ground at instant $t$ is defined as the scalar product of the integral of the reaction impulsive force $\mathbf{F}^{heel_2}\delta(t-\tau)$ in the time interval $t^- \div t^+$ with the corresponding partial velocities

$$I_r \approx \mathbf{v}_r^{heel_2}(t^-)^T \cdot \mathbf{F}^{heel_2}, r = 1, \cdots, n, \tag{27}$$

the *generalized momentum* is defined as partial derivative of the kinetic energy $K$ with respect to the r-th generalized speed

$$p_r(t) = \partial K / \partial u_r, r = 1, \cdots, n, \tag{28}$$

then, Kane proves that

$$I_r \approx p_r(t^+) - p_r(t^-). \tag{29}$$

Indicate the matrices

$$\mathbf{V}^{heel_2} = (\mathbf{v}_1^{heel_2}(t^-) \cdots \mathbf{v}_n^{heel_2}(t^-)) \tag{30}$$

$$\mathbf{P} = \{\partial p_i(t^-)/\partial u_j\}, i, j = 1, \cdots, n \tag{31}$$

of vectors of partial velocities, and of partial derivatives of $p_r(t)$ with respect to the generalized speeds, and the vectors

$$\mathbf{I} = (I_1 \cdots I_n)^T = \mathbf{V}^{heel_2{}^T} \cdot \mathbf{F}^{heel_2} \tag{32}$$

$$\mathbf{u}(t) = (u_1(t) \cdots u_n(t))^T \tag{33}$$

$$\mathbf{v}^{heel_2}(t) = \mathbf{V}^{heel_2} \cdot \mathbf{u}(t) \tag{34}$$

of generalized impulses, of generalized speeds, and the $heel_2$ velocity, respectively.

Then, from Equation (27) to Equation (31), taking into account Equations (32) and (34), and considering that $\mathbf{v}^{heel_2}(t^-)$ is given and $\mathbf{v}^{heel_2}(t^+)$ is zero, assuming non-slipping condition and inelastic collision, the following system of equations is solved to derive the unknown $\mathbf{F}^{heel_2}$ and $\mathbf{u}(t^+)$:

$$\begin{bmatrix} -\mathbf{P}\mathbf{u}(t^-) \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{v}^{heel_2}(t)^T & -\mathbf{P} \\ 0 & \mathbf{v}^{heel_2}(t) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{F}^{heel_2} \\ \mathbf{u}(t^+) \end{bmatrix} \tag{35}$$

On this respect it is worth a last remark on the controller. It will be a hybrid system, switching between two phases, as well as the biped robot. In single stance all its control components will be controlled in position, in double stance, however, the system will be over-actuated. The redundant components are, then, advantageously exploited to control reaction forces, as described in [2]. So, the collision will be easily accomodated by anticipating the switching of the control to the double stance, of a time compatible with the actuator response, e.g., exploiting ultrasound sensors under the feet. Then the control of reaction forces by the torques $\boldsymbol{T}_{aux}$ of Equation (22) in double stance will offer the needed compliance at the time of the collision. This is especially significant for accomodating the foot at the correct hight and orientation in the case of uneven ground floor.

### *3.4. The Stride in a Gait Cycle*

A stride in a gait cycle, when walking, is composed of two main phases (double and single stance) and several sub-phases [18–20]. Figure 3 shows an example of one step with the major events.

While the sub-phases defined by the physiologists cannot be adopted directly here, as they don't refer to different mechanical configurations, the event will be used in the paper.

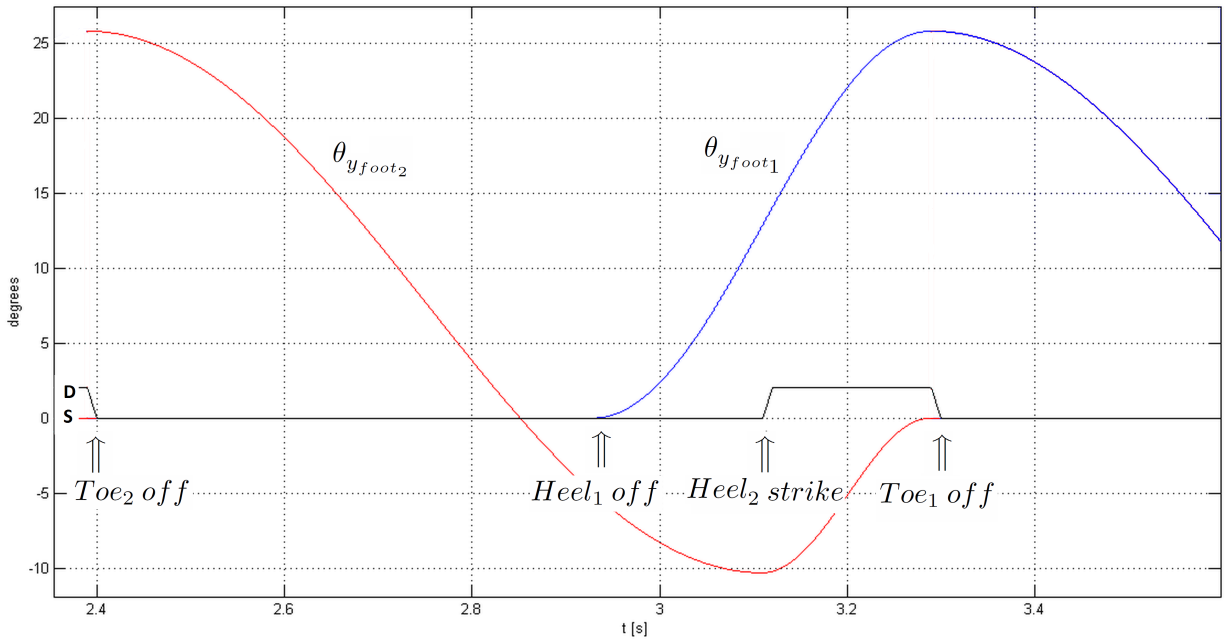


**Figure 3.** behaviour of the $\theta_{y_{foot_i}}$ and the major events during a step.

For simplicity in the presented approach, system is switching between the two main phases only.

**Double stance** In the double stance model the two feet are hinged to the ground $foot_1$ at $toe_1$ and $foot_2$ at $heel_2$. During the stance, starting from the event $Heel_2\ strike$ from the previous single stance, followed by $Foot_2\ flat$ and terminating with $Toe_1\ off$ (For simplicity here these last two events are set at the same instant.), the weight is transferred from $foot_1$ to $foot_2$.

**Single stance** In the single stance model the supporting $foot_1$ is hinged at $toe_1$. The phase starts with the supporting $foot_1\ flat$ at the event $Toe_2\ off$, followed by $Heel_1\ off$ and ending with $Heel_2\ strike$.

If during the first sub-phase the posture is controlled by the support of the foot flat on the ground, during the second the active torque on the $toe_1$ joint is needed.

Note that there are two symmetrical models for each stance. We define them as *modes*: right (the conventional one of Figures 1 and 2) and left mode. They differ by a parameter that assumes the value $+1$ for the right mode and $-1$ for the left mode, and by exchanging the two sets of limb indexes in the left mode. During the gait, in order to alternate right stance phase and left stance phase, we adopted the following convention: when switching from single to double stance the same mode is maintained, from double to single stance the mode is switched.

## 4. Building the Simulator

### *4.1. The Autolev Symbolic Computation*

In Autolev the description of holonomic or non-holonomic systems is identical. Along with the definition of the structure of the multichain (bodies, points, segment *COGs*, masses, inertias, kinematical relationships). For the dynamics, input forces/torques, states and output variables have to be specified. States are however divided in positions and velocities, linked by kinematical differential equations:

**State 1** the generalized coordinates;

**State 2** the generalized speeds;

**Kinematical differential equations** the relationships between coordinate derivatives and speeds (Equation (6);

**Input** all forces and torques (comprehensive of those returned by the constraints) with reference, respectively, to the points or bodies of application, to be considered external, hence to be possibly controlled;

**Output** the desired variables in output, e.g., joint angles and speed, *COG* and *ZMP* coordinates, swing foot coordinates, Jacobian matrices, linearizations matrices of the differential equations, reaction forces/torques, matrices of partial derivatives of reaction forces/torques with respect to input control torques, partial velocities, generalized impulse, generalized momentum, *etc*. In addition, for non-holonomic systems it is required to declare:

- the *non-holonomic* constraints on the generalized speeds (see Equation (12));
- the selected dependent generalized speeds;
- forces and torques, in the previous declaration, resulting from the constraints, therefore, not to be considered as external input, but internally derived by the simulator.

The Autolev environment interprets the configuration and declarations and generates the dynamical equations and espressions for the output variables (individuals, vectors or matrices as defined), splitting for efficiency in an array of internal variables (called $Z$) the intermediate results. As output, the source code in C, Fortran or Matlab and an initialization file are returned. Two group of files are then generated, one for each stance.

#### 4.1.1. The Generated Code

The code generated for simulation is composed of the variable declarations, a main program and a set of routines. Among them, a group of service routines are identical for any configuration (reading input parameters and initial conditions, formatting output variables, implementing Runge-Kutta numerical integration), vice versa, three functions are specific of each configuration:

**Initialization** parameters and state initialization (setting parameters and initial conditions);

**State derivative computation** a function that computes the state derivative (generalized coordinates and generalized speeds), given the control input, to be called at each phase initializations and at each sampling period during integration by the Runge-Kutta routine in order to update the intermediate expressions, variables and the state;

**Output computation** a function to calculate at each sampling period any other intermediate expression or variable dependent on the state, state derivative, and input, declared as output in the program. These samples are also saved in a file at the end of the simulation.

For control, a module (the controller) has to be added to the simulation code that sets before each sampling time, the values of forces/torques declared as control input.

### *4.2. The Object Oriented Composition of the Program*

The C code returned by Autolev, with a few simple modifications, that we automated with a parser, can be embedded in an object oriented programming framework originating one class for each phase of the system, all inheriting from a common base parent, as shown in Figure 4. The base parent contains the declaration of the common variables (intermediate expressions $Z$, parameters, generalized coordinate, generalized speeds and their derivatives, and the *mode* variable $(+1, -1)$, implements the common service routines, and declares as virtual one protected method for computing the derivative of the state, and two public methods: one for initialization in the right and left mode, and one for returning output variables. Each one of the phase classes implements the three virtual methods.

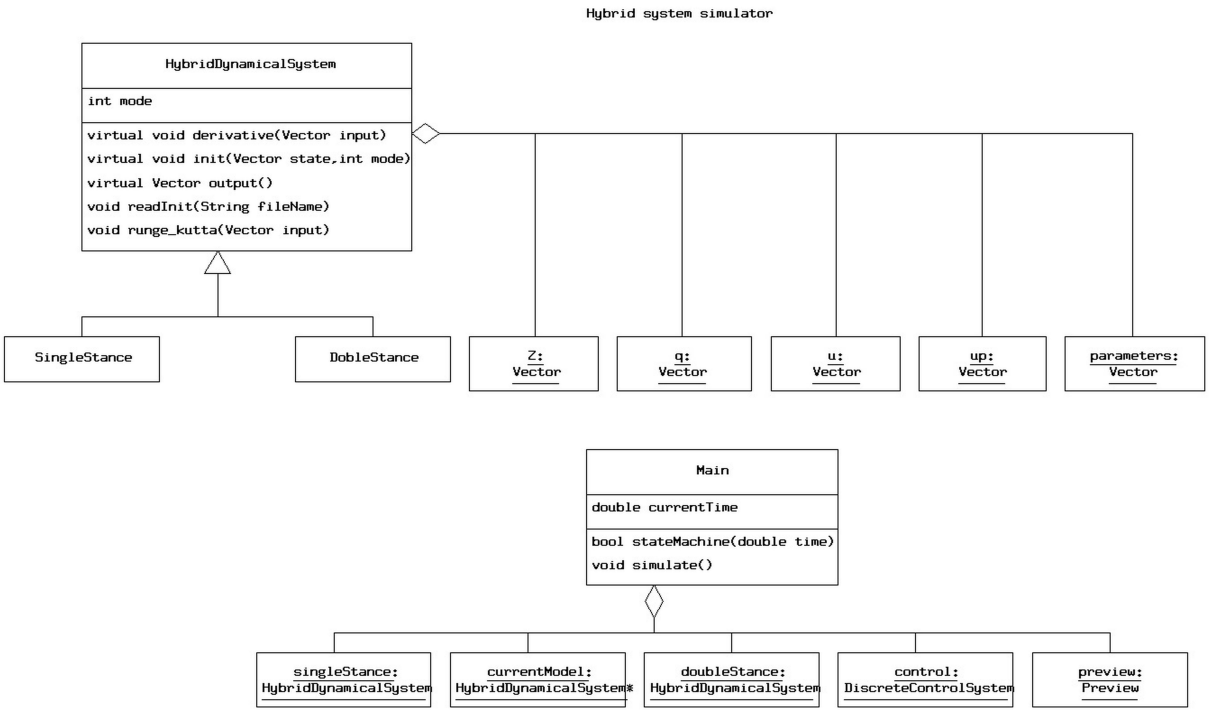


**Figure 4.** The object model of the simulator in UML notation [21].

#### 4.2.1. The Main Class

The main class has as member variables one instance of each phase class (*SingleStance*, *DoubleStance*) of the system, a pointer to the *currentModel*, a *Preview* to generate the reference trajectories, and a *Control* class to generate the control inputs. It runs the simulation from initial to final instants with two threads: one, event driven ($Toe_1off$, $Heel_2strike$), controlling the phases

and modes through a state machine, depicted in Figure 5, the other with a given fixed sampling time closing the control loop. Samples of Autolev Code Fragments for building the model and generate the C++ code are presented in Appendix A.

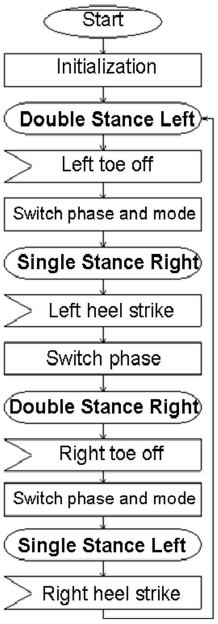


**Figure 5.** The state machine of the simulator in SDL notation [21].

The Preview

The preview generates the nominal reference trajectories of the robot in the Cartesian space, composed by the *COG* and *COP* trajectories in the frontal and sagittal planes, the position and angles of the swing foot in single stance, the rotation angles of both feet in double stance.

The Control

The control is a sampled time hybrid dynamical system: at each time period it receives reference and feedback signals and returns the control, updating its state. It switchs configuration at each phase of the model.

## 5. Conclusions

In this paper, a modelling approach and a computer simulator for biped robotics, outlining the advantages of adopting the Kane's method to describe the dynamics, has been proposed. The simulator has been developed in Object Oriented programming with the C++ language by exploiting, for the software code of the mathematical expressions, the symbolic dynamic manipulator *Autolev* that complements the Kane's method. The Kane's method together with *Autolev*, with the introduction of intermediary variables to avoid repetions, assure tha maximum efficiency of computation.

As a case study, a 3D standard 12 DOF biped configuration has been considered, however it is clear that the approach can be advantageously applied to different configurations.

Especially attractive is the handling of non-holonomic constraints, when used for rehabilitation, with the possibility to embed in the model different kinds of constraints, as mentioned in Section 2, such as interaction with a chair or a pair of crutches, a thrust, *etc*. In fact the original motivation of this work has been the development of a lower limb haptic exoskeleton for postural rehabilitation, that is actually being currently built under Italian Government grants, and a sit-to-stand exercise implementation [22].

The object oriented framework adopted to develop the simulator is based on the $G++$ programming environment by SYCO [21].

Examples of controls based on this model can be found in [2] and in part 2 and 3 of this paper.

**Acknowledgments:** This research has been partially supported by MIUR the Italian Ministry of Instruction, University and Research and the Region Piedmont.

**Author Contributions:** The authors contributed equally to this work.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A. Code Fragments

The following listing presents a sample of the input code of Autolev to generate the Kane's dynamical equations, the *COG* coordinates, the matrices of the linearized model, the *COG* Jacobian matrix, the sensitivity of reaction forces/torques from joint input torques. The single stance configuration has been considered here.

```
% Declaration of the configuration variables
Variables   xthetaAnkle1', ythetaAnkle1', ythetaAnkle1', ythetaAnkle2',
zthetaAnkle1',zthetaAnkle2',thetaKnee1',thetaKnee2', xthetaHip1',
xthetaHip2', ythetaHip1', ythetaHip2', xthetaFoot1', ythetaFoot1',
xFoot1', yFoot1',zFoot1'
% Declaration of the motion variables
MotionVariables' uxAnkle1', uxAnkle2', uyAnkley1', uyAnkley2', uzAnklez1',
uzAnklez2', uKnee1', uKnee2', uxHip1'', uxHip2',  uyHipy1',uyHipy2',
uxFoot1',uyFoot1',vxFoot1',vyFoot1', vzFoot1'
% Declaration of the vector of reaction forces/torques to be computed
% and returned by by Autolev
Zee_Not = [FxFoot1; FyFoot1; FzFoot1; TxFoot1; TzAnkle1; TxAnkle2; TzAnkle2]
% Command that generates the Kane dynamical equations
% Zero is the vector contianing the differential equations eq. (11)
Zero = Fr() + FrStar()
% Command that resolves nonholonomic constraints:
% it generates the reduced order dynamical equations
% and returns the reaction forces/torques
Kane(Zee_Not)
% The COG coordinates: cm returns the COG vector
% dot performs the scalar product of the vector with the x,y and z axis
cogx = dot(cm(No,foot1,foot2,leg1,Thigh1,leg2,Thigh2,Hat),N1>)
cogy = dot(cm(No,foot1,foot2,leg1,Thigh1,leg2,Thigh2,Hat),N2>)
cogz = dot(cm(No,foot1,foot2,leg1,Thigh1,leg2,Thigh2,Hat),N3>)
% The time derivatives (Dt) of the COG coordinates are computed
cogxp = Dt(cogx)
cogyp = Dt(cogy)
cogzp = Dt(cogz)
% The vectors of the configuration variables,
% of the generalized spees and accelerations are defined
thetas = [thetayFoot1,thetayAnkle1,thetaKnee1,thetayHip1,thetayHip2,thetaKnee2
,thetayAnkle2,thetaxAnkle2,thetaxHip1,thetaxHip2]
u = [uyFoot1,uyAnkle1,uKnee1,uyHip1,uyHip2,uKnee2,uyAnkle2,uxAnkle1,uxHip1,uxHip2]
up = [uyFoot1',uyAnkle1',uKnee1',uyHip1',uyHip2',uKnee2',uyAnkle2',uxAnkle1'
,uxHip1',uxHip2']
% The vector of input joint torques is defined
torques = [TyFoot1,TyAnkle1,TKnee1,TyHip1,TyHip2,TKnee2,TyAnkle2,TxAnkle1,TxHip1
,TxHip2]
% COG Jacobian has been computed making the partial derivatives (D) of COG velocity
% with respect to the generalized speeds
Jcog = D([cogxp;cogyp;cogzp],u)
% The matrices of the linearized dynamical model are computed as partial derivatives
% of the vector of the differential equations
% the inertia matrix with respect to the accelerations
IM = D(Zero,up)
Zero0 = evaluate(Zero, u = 0, up = 0,torques = 0)
```

```
% Linearization of gravitational forces with respect to the positions
K = D(Zero0,thetas)
% The input matrix with respect to the input joint torques
B = D(Zero,torques)
% The matrix of linear relationship between reactions forces/torques
% with respect to inputs
Jforce_torque = D(Zee_Not,torques)
% The command partials returns the partial velocities of heel2
V = partials(V_Heel2_N>)
% Momentum returns the angular momentum
P =  momentum(generalized)
```

Next is a fragment of the Autolev output; Z2545, Z2535, ... are intermediary variables generated automatically for efficiency from the environment. The first line is one component of the vector of the dynamical differential equations, the second line is the expression of the vertical reaction forces on foot 1.

```
-> (3077) ZERO[1] = Z2545 + Z2535*uxAnkle1' + Z2536*uyAnkle1' + Z2537*uyAnkle2'
+ Z2538*uKnee1' + Z2539*uKnee2' + Z2540*uxHip1' + Z2541*uxHip2' + Z2542*uyHip1'
+ Z2543*uyHip2' + Z2544*uyFoot1'

(...)

-> (3061) FzFoot1 = Z2405*uxAnkle1' + Z2406*uyAnkle1' + Z2407*uyAnkle2'
+ Z2408*uKnee1' + Z2409*uKnee2'+ Z2410*uxHip1' + Z2411*uxHip2'
+ Z2412*uyHip1'+ Z2413*uyHip2' + Z2414*uyFoot1' - Z2472
```

A sample of the *simulate* method of the class *Main*, controlled by the state machine of Figure 5. *singleStance.in* and *doubleStance.in* are initialization files automatically generated from Autolev.

```
singleStance.readInit("singleStance.in");
doubleStance.readInit("doubleStance.in");
currentModel = &doubleStance;
int mode = currentModel->getMode();
currentTime = currentModel->getTime();
samplingTime = currentModel->getSamplingTime();

control.setSamplingTime(samplingTime);
currentModel->init(currentModel->getState(), mode);
control.init(currentModel->getState(), mode, preview.reference(currentTime)
, currentModel->getPhase());

Vector modelInput = control.output();
currentModel->derivative(modelInput);
currentModel->output(); //print initial values

while(currentTime < Tend)
{
        currentModel->runga_kutta(modelInput,currentTime);
        currentTime = currentModel->getTime();

        if(stateMachine(currentTime))  // sense events of  heel_strike and toe_off
        {
                Vector state = currentModel->getState();
                int mode = currentModel->getMode();
                if (currentModel == &singleStance)
                {
                        currentModel = &doubleStance;
                        currentModel->init(state,mode);
                }
                else
                {
```

```
                        currentModel = &singleStance;
                        currentModel->init(state,mode *= -1);
                }
                control.init(state, mode, preview.reference(currentTime)
                , currentModel->getPhase());
                modelInput = control.output();
                currentModel->derivative(modelInput);
                currentModel->output();
        }
        else
        {
                Vector modelOutput = currentModel->output();
                control.next(modelOutput, preview.reference(currentTime));
                modelInput = control.output();
        }

}
```

The last fragment is the Autolev generated implementation in C code of the previous vertical reaction force at foot 1:

```
ForcezFoot1 = z[2405]*uAnkle1p + z[2406]*uAnkley1p + z[2407]*uAnkley2p +
z[2408]*uKnee1p + z[2409]*uKnee2p + z[2410]*uHip1p + z[2411]*uHip2p +
z[2412]*uHipy1p + z[2413]*uHipy2p + z[2414]*uFooty1p - z[2472];
```

## References


1. Menga, G. Team Isaac: Academic Research Project on Robotic Humanoids. Available online: http://www.isaacrobot.it/ (accessed on 17 March 2016).
2. Menga, G.; Ghirardi, M. Lower Limb Exoskeleton with Postural Equilibrium Enhancement For Rehabilitation Purposes. Internal report DAUIN - Politecnico di Torino. 2015. https://dl.dropboxusercontent.com/u/26129003/SUBM1.pdf (accessed on 17 March 2016).
3. MSC Software, MSC Adams Multibody Dynamics Simulation. Available online: http://www.mscsoftware.com/Products/CAE-Tools/Adams.aspx (accessed on 17 March 2016).
4. Hurmuzlu, Y.; Genot, F.; Brogliato, B. Modeling, stability and control of biped robots—A general framework. *Automatica* **2004**, *40*, 1647–1664.
5. Heemels, W.P.M.H.; Brogliato, B. The complementarity class of hybrid dynamical systems. *Eur. J. Control* **2003**, *9*, 322–360.
6. Siciliano, B.; Oussama, K. *Springer Handbook of Robotics*; Springer: Berlin, Germany; Heidelberg, Germany, 2008.
7. Kane, T.R.; Levinson, D.A. *Dynamics: Theory and Applications*; McGraw-Hill: New York, NY, USA, 1985.
8. Gillespie, R.B. Kane's equations for haptic display of multibody systems. *Haptics-e* **2003**, *3*, 144–158. Available online: http://www.haptics-e.org (accessed on 17 March 2016).
9. Mitiguy, P. MotionGenesis: Advanced Solutions for Forces, Motion, And Code-Generation. Available online: http://www.motiongenesis.com/ (accessed on 17 March 2016).
10. Azevedo, C.; Espiau, B.; Amblard, B.; Assaiante, C. Bipedal locomotion; toward a unified concept in robotics and neuroscience. *Biol. Cybern.* **2007**, *96*, 209–228.
11. Azevedo, C.; Amblard, B.; Espiau, B.; Assaiante, C. *A Synthesis of Bipedal Locomotion in Human and Robots*; Technical Report; Research Report 5450; INRIA: Sophie Antipolis Cedex, Nice Cedex, France, 2004.

    Available online: https://hal.inria.fr/inria-00070557 (accessed on 17 March 2016).
12. Wieber, P. Constrained Stability and Parameterized Control. In *Biped Walking*, Proceedings of the International Symposium on Mathematical Theory of Networks and Systems, Perpignan, France, 19–23 June 2000.
13. Wieber, P. On the Stability of Walking Systems. In Proceedings of the International Workshop on Humanoids and Human Friendly Robotics, Tsukuba, Japan, 11–12 December 2002.

14. Kumar, R.; Handharu, N.; Jungwon, Y.; Gap-soon, K. Hybrid toe and heel joints for biped/humanoid robots for natural gait. In Proceedings of the International Conference on Control, Automation and Systems, Seoul, Korea, 17–20 October 2007.
15. Ezati, M.; Toosi, K.; Khadiv, M.; Moosavian, S. Dynamics modeling of a biped robot with active toe joints. In Proceedings of the 2014 Second RSI/ISM International Conference on Robotics and Mechatronics, Tehran, Iran, 15–17 October 2014.
16. Ezati, M.; Khadiv, M.; Akbar, M.S.A. Optimal gait planning for a biped robot by employing active toe joints and heels. *Modares Mech. Eng.* **2015**, *15*, 69–80.
17. Bajodah, A.H.; Hodges, D.; Chen, Y. Nonminimal generalized Kane's impulse-momentum relations. *J. Guid. Control Dyn.* **2004**, *27*, 1088–1092.
18. Ounpuu, S. The biomechanics of walking and running. *Clin. Sports Med.* **1994**, *13*, 843–863.
19. Saunders, J.B.; Inman, V.T.; Eberhardt, H.D. The major determinants in normal and pathological gait. *J. Bone Joint Surg.* **1953**, *35-A*, 543–558.
20. Patton, J. *Normal Gait Part B of Kinesiology*; Technical Report, 2002. Available online: http://www.smpp.northwestern.edu/~jim/kinesiology/partB_GaitMechanics.ppt.pdf (accessed on 17 March 2016).
21. Menga, G. G++ Programming Environment, SYCO. 2008. Available online: www.syco.it (accessed on 17 March 2016).
22. Bahar, B.; Miskon, M.F.; Bakar, N.A.; Shukor, A.Z.; Ali, F. Path generation of sit to stand motion using humanoid robot. *Aust. J. Basic Appl. Sci.* **2014**, *8*, 168–182.