Development of a Dynamic Simulator for a Compliant Humanoid Robot Based on a Symbolic Multibody Approach

Houman Dallali, Mohamad Mosadeghzad, Gustavo A. Medrano-Cerda, Nicolas Docquier, Petar Kormushev, Nikos Tsagarakis, Zhibin Li, Darwin Caldwell

Abstract—**This paper reports on development of an open source dynamic simulator for the COmpliant huMANoid robot, COMAN. The key advantages of this simulator are: it generates efficient symbolic dynamical equations of the robot with high degrees of freedom, it includes a user-defined model of the actuator dynamics (the passive elasticity and the DC motor equations), user defined ground models and fall detection. Users have the freedom to choose the proposed features or include their own models. The models are generated in Matlab and C languages, where the user can leverage the power of Matlab and Simulink to carry out analysis to parameter variations or optimization and also have the flexibility of C language for real-time experiments on a DSP or FPGA chip. The simulation and experimental results of the robot as well as an optimization example to tune the ground model coefficients are presented. This simulator can be downloaded from the IIT website [1].**

INTRODUCTION

DEVELOPMENT of a realistic and mathematical model of humanoid robots motion plays a key role in design and testing of controllers and trajectory generators. The desired simulator for COMAN should include the direct and inverse dynamics, forward and inverse kinematics, actuator dynamic models, series elastic elements, ground model with friction, and good computational speed. Among the simulators for humanoid robots Open Dynamic Engine (ODE) [2], Webots [3], Open HRP [4], SL [5], Adams [6], RoboWorks [7], SimMechanics, MapleSim [8], player/stage, Gazebo, SAI (Simulation & Active Interface) [9] developed in Stanford University and the simulators developed by Honda and Sony are the most well known. However, some of these simulators are commercial or not fully open source which is problematic when testing and debugging the controllers, since not all their details are accessible or customizable to suit a particular robot.

Adams is a powerful commercial multibody dynamics software which is widely used in industry and some universities. However it is not open source, which makes it difficult to share the simulator among research groups who work in humanoid robotics. Moreover, Adams generates the equation of motion or kinematics numerically as opposed to the symbolic approach. OpenHRP and SL are strong simulation tools to validate control algorithms for stiff robots since they support rigid body dynamics, but they miss the significant actuator dynamics present in flexible joint robots such as COMAN. ODE is an open source library created for modeling rigid body robots, but does not satisfy all the contact dynamic requirements that are needed in creating a realistic walking simulator. Also, important dynamic and kinematic information such as Jacobian and linearization information has to be computed manually. Webots is a commercial robotic simulation tool which provides a user interface to the ODE.

A strong multi-body modeling tool is Robotran [10] which is used to model COMAN due to the following advantages and based on the previous work presented in [11]. It generates the mathematical equations of the robot symbolically as opposed to numerical models which are less efficient and more sensitive to numerical errors. Then, in the Robotran approach, the implementation of application specific features is left to the user. However, to assist the user, Robotran framework provides predefined templates to add actuator dynamics, constraints, external forces on any point on the robot, as well as sensors with symbolic expressions to acquire kinematic information such as Jacobian, linear and angular position, velocity and acceleration of any point of interest on the multibody system. All equations are generated in both C and Matlab languages to leverage the power of Matlab in the development stage and the flexibility of C in the final dissemination and real-time experimentation stage. The symbolic models and dynamic parameters are stored separately which makes it convenient for updates and modifications. The parameters of COMAN used in the simulations are obtained from the CAD data and catalogue of the actuators. System identification is not in the scope of this paper.

The main contribution of this paper is the development of a compliant humanoid robot simulator based on efficient symbolic dynamic models. The symbolic models are generated once and simplified with up to 30% fewer equations using Robotran engine. The complete open source code of the simulator is available for download from the IIT website [1] for research purposes under GNU license. The

H. Dallali, M. Mosadeghzad, G. A. Medrano-Cerda, Z. Li, N. Tsagarakis and D. G. Caldwell are with the Dept. of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163, Genoa, Italy (emails: [houman.dallali, mohamad.mosadeghzad , gustavo.cerda, petar.kormushev, nikos.tsagarakis, zhibin.li, darwin.caldwell]@iit.it). M. Mosadeghzad is also affiliated with the faculty of engineering, university of Genoa.

N. Docquier is with the centre for research in mechatronics (CEREM), universite catholique de Louvain. Place du Levant 2, bte L5.04.02, B-1348 Louvain-la-Neuve, Belgium (email: nicolas.docquier@uclouvain.be).

978-1-4673-1388-9/13/$31.00 ©2013 IEEE

simulator is developed in both Matlab and Simulink. The Matlab/Robotran version uses m-files and ODE solvers. On the other hand, the Simulink/Robotran version uses Simulink S-functions with all direct dynamics and actuator models compiled in C, as a mex file which speeds up the simulations considerably. This simulator can use all the power of Matlab to carry out analysis on the performance and optimization on the control software and all design parameters. Furthermore, the generated C code can be executed in Windows or Linux operating systems independent from Matlab and Simulink for real-time experiments on a DSP or FPGA.

This paper is organized as follows. Section II gives an overview of COMAN, describes the mathematical walking model and the Robotran features as well as the method of developing the floating base walking model. In addition, the linear and nonlinear ground contact models and integration of the equations of series elastic actuators into the simulator are described. Section III presents the simulation and the experimental results. Finally, the conclusions and future work are discussed in Section IV.

COMAN'S MODEL BASED ON ROBOTRAN

A. Overview of COMAN Robot

COMAN is a humanoid robot (Fig. 1), powered by series elastic actuators, and is being developed within the AMARSI European project [12], at the Italian Institute of Technology (IIT) as a derivative of the original iCub, and cCub [13] which added passive compliance in the major joints of the legs. The use of passive compliance will provide shock protection, robust locomotion, safer interaction and potentially energy efficient locomotion.

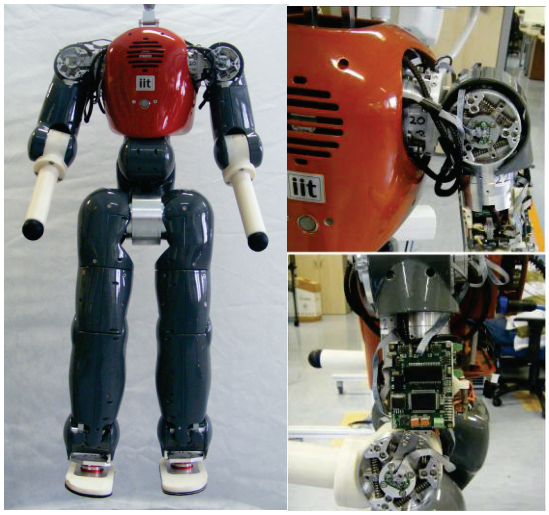

Fig. 1. COMAN humanoid robot.

Currently COMAN has 23 DoF, where the pitch joints in the legs, the waist, the shoulders and the shoulder roll joints have passive compliance. In addition, the robot uses brushless DC motors and harmonic drives, controlled with PID position control, which are modeled in the Robotran simulator. Further details about previous prototype of COMAN, cCub are available in [13] with the major kinematic difference that the ankle and waist roll-pitch order is swapped.

B. Floating Base Humanoid Model

The proposed COMAN dynamic simulator is based on the floating base representation of legged robots as proposed in the literature [14-16]. A floating base has its base body free to move rather than being fixed in space. The free motion of the base is represented by a 6 DoF attached between the humanoid robot and the world inertial frame to describe the free motion of the humanoid with respect to the inertial frame (Fig. 2). The 6 DoF consist of three translational joints and three rotational joints about the XYZ axis. This formalism unifies all the phases of legged locomotion, including single support, double support and flight phase, as well as the falling phase into one single model, and simplifies the simulation models by removing the switches between various phases of walking. Since the floating base joints are un-actuated, the ground model has a significant role in the robots balance and locomotion. The ground models are explained in Section II.D.

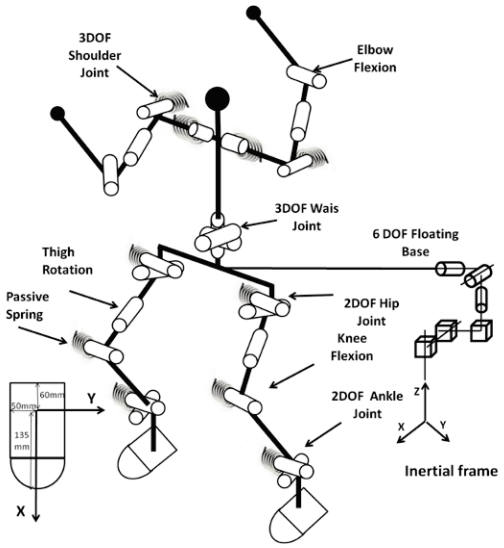

Fig. 2. COMAN's floating base kinematic model.

C. COMAN Model in the Robotran Environment

Robotran is a general purpose multibody software developed at the Université catholique de Louvain [10]. It relies on a symbolic approach dedicated to mechanical systems. The Robotran engine concentrates on the symbolic generation of the motion equations, which is a common task to any kind of mechanical system. Then, in the Robotran philosophy, the specific work such as writing the constitutive law of an actuator or implementing a ground contact model are delegated to the user.

Robotran has three main components, the graphical user interface (MBsysPad), a symbolic equation generator

(MBsysTran) and a simulation environment for Matlab/Simulink (MBsysLab). The kinematic and dynamic parameters of the robot are entered via the java based MBSysPad editor (Fig. 3), and stored in an XML file.

This editor relies on a 2-D diagram representation of the system as illustrated in Fig. 3, which gives a complete and straightforward overview of the 3-D mechanical model's topology. The rectangular shapes represent each body of the robot as a point mass with inertia, centre of mass and suitable dimensions. Robotran defines six simple joints with 1 DOF: three rotational joints about XYZ axis represented by R1, R2, R3 and three translational joints along XYZ are represented by T1, T2, and T3. These joints can be combined in many ways for introducing more complex joints (universal joints, spherical joints, etc). For instance, the six DoF floating base placed between the base (the gray rectangle in Fig. 3) and the waist of the robot is obtained by combining the six simple joints.

Specific points on the system are introduced using *anchor points* attached to a given body and depicted by arrows in the 2-D diagram (Fig. 3). An anchor point is defined by three constant coordinates with respect to the body fixed frame. Then, they are used to define the position of joints, sensors or external forces. Each sensor is denoted by the symbol **S** which gives the symbolic kinematic information such as position, orientation, Jacobian matrix, linear/angular velocities and linear/angular accelerations of a given body with respect to the base frame.

Once the system topology and parameters have been completely introduced in MBsysPad, the symbolic equation generator is accessed online via the Robotran web server. The symbolic equations are generated in a few seconds, either in Matlab language or in C language and they are then downloaded to the project folder as a set of Matlab M-files or C-files. Robotran can generate both direct dynamics (calculation of the trajectory for given actuator torque) and inverse dynamics (calculation of the actuator torque for a given trajectory) models. Once the models are generated, the symbolic files are completely independent from the symbolic engine and can be used in any simulation or control environment, or transferred to a DSP or FPGA, which make the model very portable. Subsequently, the user has to implement the relevant actuator dynamics or external forces. This approach makes the software really open and flexible instead of reducing the possibilities to a limited set of predefined functionalities.

For the simulation, the Robotran framework comes with the MBsysLab environment which provides functionalities for performing the dynamic analysis (equilibrium, time simulation, modal analysis, etc.) in Matlab and/or Simulink. The model can also be coupled with the control facilities of Simulink. A 3D visualization tool is available in MBsysPad, which can be used to animate the result of a simulation as shown in the accompanying video. Details of how modules of Robotran are interconnected are given in the Appendix (Fig. 10).

D. Ground Contact Model

The ground models are introduced in COMAN simulator using linear and nonlinear spring-damper models with realistic friction to allow slippage on the floor as well as producing normal ground reaction forces. This is compared in Section III with the experimental data.

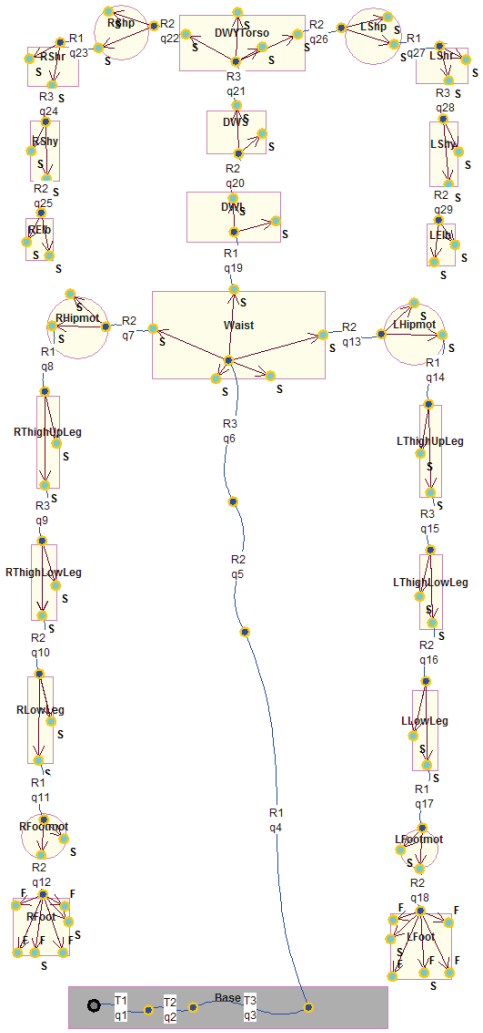

Fig. 3. Floating base model of COMAN in Robotran.

In the tangential directions X and Y, both models have two regimes. One is the sticking (coulomb friction) mode and the other is the sliding mode. The resolution between the two modes is done as follows. Assuming that the foot is in sticking mode the tangential friction force are computed and compared against the μF_z where μ is the ground friction coefficient (for instance $\mu=0.7$) and F_z is the normal force. If the sticking force is greater than μF_z (tangential force is outside the friction cone), the friction is set equal to μF_z and the contact point is allowed to slip, otherwise the tangential force is kept as initially computed using the spring-damper model. In linear model, the normal force is calculated based on linear spring and damper:

$$F_z = \begin{cases} 0 & z > 0 \\ K z + D \dot{z} & z \le 0 \end{cases} \quad (1)$$

where z denotes the contact point penetration in the ground. However, in nonlinear model, the normal force is formulated as proposed in [17]:

$$F_z = \begin{cases} 0 & z > 0 \\ K_n z^{\frac{3}{2}} + D_n z^{\frac{1}{2}} \dot{z} & z \le 0 \end{cases} \quad (2)$$

where K_n and D_n are nonlinear spring and damper coefficients which are obtained using an integral over the contact surface area according to Hertz's theory. These models are programmed as external forces acting on the feet bodies (user_ExtForces template of Robotran package, see Fig. 10 in the Appendix). It should be noted that this method is one way of modeling the contacts, while the user can alternatively model the contacts using constraints [18]. Also, the user can choose to model soft or hard contacts, depending on his application. In Section III.C, a reinforcement learning based optimization, as proposed in [19], is used to automatically tune the linear ground model based on the experimental data.

E. Actuator Dynamics

Actuator dynamics have a significant effect on the overall dynamics of COMAN which should be modeled. This can be added to the multibody model in Robotran as additional user derivatives (ordinary differential equations introduced in the user_Derivatives template in the Robotran framework). Furthermore the coupling torque between the motors and the multibody system, due to transmission is added to the mechanical model as a joint torque (user_JointForces template). This is given in (5). The overall model of series elastic actuator connected to the multibody model system is:

$$M(q)\ddot{q} + C(\dot{q}, q) = \tau_L \quad (3)$$

$$J\ddot{q}_m + D\dot{q}_m + \tau_L = V_T u \quad (4)$$

$$\tau_L = D_s\left(\dot{q}_m - \dot{q}\right) + K_s\left(q_m - q\right) - C\dot{q} \quad (5)$$

where, J is motor inertia, D includes the motor back EMF constant and the rotor friction, K_s and D_s are passive stiffness of damping of all joints, C is the viscose damping on the link side, τ_L is the coupling torque between the motor and the joint, V_T is voltage to torque ratio, and u is the control voltage signal calculated using a PD controller. The left hand side of (3) is generated symbolically with Robotran which gives the nonlinear mass-inertia matrix, coriolis and gravity matrices (mbs_DirDyna function in Fig. 10, either in Matlab or C code).

F. Collision and Fall Detection

The Robotran's sensor feature and event detection in Matlab are combined in order to detect a fall and stop the simulation. The sensor which is placed on the waist of the robot is called by Matlab ODE solver at each time step to monitor the height of the robot and an event related to falling is produced when the waist height is lower than a certain value, for instance 30 (cm).

In addition, basic self collision detection can be added to the program by using the position information of the sensors which are placed at the centre of gravity of each body (Fig. 3). A neighborhood of each centre of gravity can be monitored by Matlab ODE solver to produce an event in the case of self-collision. In the Section III, simulation and experimental results of COMAN are presented.

SIMULATION AND EXPERIMENTAL RESULTS

In this section, initially a single joint test (on right knee) is done to compare the control signals. In addition, the results of simulating a lateral sway are presented and compared with experimental data from the robot. The corresponding tracking errors and ground reaction forces are presented. The integration method used is ODE45 in Matlab with a variable step time.

G. Control Signal Comparison

Intrinsic characteristic of joints namely passive stiffness, viscose friction, and stiction are estimated by simulator using a single joint (right knee) experiment. In this test, COMAN is suspended in the air by ropes and a smooth reference position from 0 to 90 degree (bending a knee) is sent to the joint, as shown in Fig. 4. The PD controller of the knee joint has a proportional value of K_p = 200 (V/rad), and derivative value of K_d =10 (V.sec/rad).

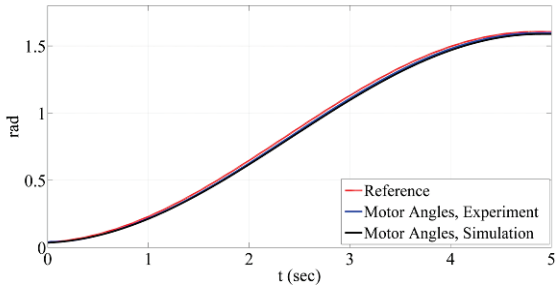

Fig. 4. A smooth trajectory sent to the right knee joint.

After adding the effects of link viscose damping (estimated as 8 (Nm.sec/rad)), damping across the spring (0.8 (Nm.sec/rad)) and stiction (estimated as 0.6 (V)), simulated voltage shows a good agreement with the experimental voltage (Fig. 4). However, still voltages in steady state do not fully match, as there is a 0.6 (V) difference. This is mainly due to the actual stiction level and the mass of the plastic covers of the robot which are not included in the simulation at this stage. In the rest of this paper, the results of the lateral sway are discussed.

H. Trajectory Tracking

Link and motor positions, velocities and tracking errors of all joints are computed from the mathematical models in the simulation. In this section, the tracking errors of right hip and

ankle lateral joints during the sway motion are shown in Fig. 6. The left hip has a mirror image of the presented results. The torques are not measured but the control voltages are generated by PD control with proportional value of K_p =150 (V/rad), and derivative value of K_d =10 (V.sec/rad), which will be applied to the tracking errors illustrated in Fig. 6.

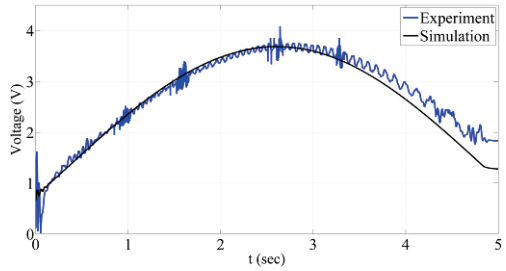

Fig. 5. The corresponding control signals of the right knee joint.

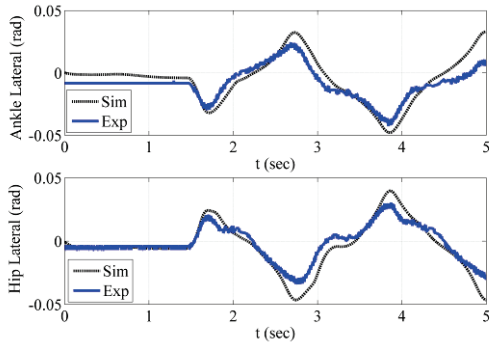

Fig. 6. Tracking errors during simulation and experiment.

I. Ground Reaction Force

As mentioned in Section II.C, two linear and nonlinear ground models are used in the simulator. The reinforcement learning based on particle filtering algorithm (RLPF) presented in [19] was used to automatically tune the stiffness and damping coefficients of the linear model. This optimization was performed in 100 trials to tune the parameters and the final result is shown in Fig. 7. The RMS error between the simulated linear ground model and the experimental data while the robot is moving is about 50.4 (N). This error is partly due to the weight of the covers and mass distribution of the robot, but more importantly due to the approximation of the foot contact with 5 single points instead of an area.

The stiffness and damping was chosen as 203600 (N/m) and 1006 (N.sec/m) according to the optimization shown in Fig. 7. The linear model produces faster simulations compared with the nonlinear model.

Simulation of the nonlinear ground model is shown in Fig. 8. It can be seen that the initial contact with the ground is damped rapidly and the overall profile of the simulation agrees with the experiment. In terms of computational time, the nonlinear model is slower. The stiffness and damping coefficients chosen for the nonlinear model are 400K (N/m) and 400 (N.sec/m). The experimental data for the ground reaction force is measured using six DoF force/torque sensors, installed under the feet of the robot.

Moreover, this simulator was used to tune walking trajectories for 15 DoF model of the robot (excluding the torso and the arms) where the reinforcement learning based on particle filtering (RLPF) algorithm was used to tune the ZMP walking trajectories. After using few hundreds of trials, several stable and dynamic walking gaits were obtained [20]. This is an example of how the simulator can benefit from the power of Matlab for analysis and optimization. As a further illustration of COMAN simulator, a walking gait was chosen and applied to the robot and the simulation which produced stable walking in both simulation and real-world as shown in the accompanying video [21], which shows the robot in actual time.

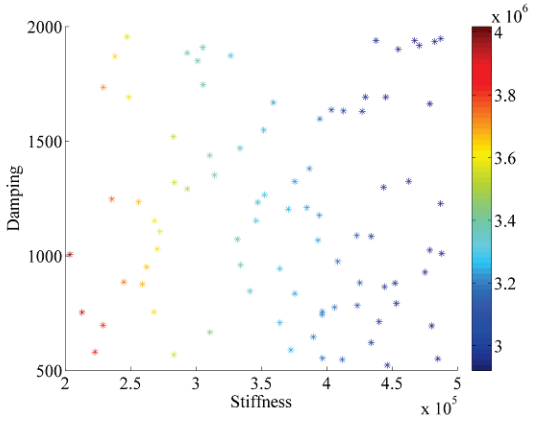

Fig. 7. Color coded stiffness and damping parameters of the linear ground model, with optimum values shown in red spectrum.

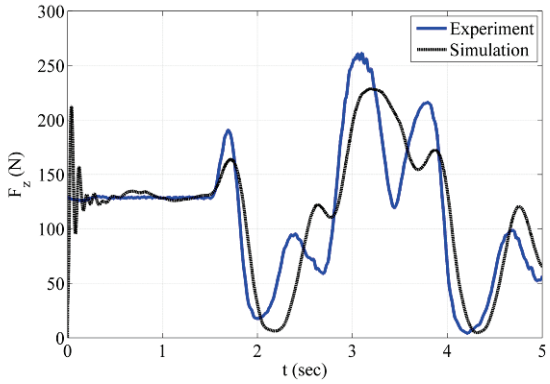

Fig. 8. Simulated nonlinear ground reaction force against experiment.

CONCLUSIONS AND FUTURE WORK

This paper presented an open source, floating base whole-body dynamic simulator for the compliant humanoid robot, COMAN including the series elastic actuator dynamics and realistic ground models. The presented results show one of the few open source and fully customizable simulators which works both in Matlab and C languages. The simulation predictions were compared with a single joint test, an

experimental sway data on COMAN and also walking tests on the COMAN prototype, cCub. This simulator can be used to test and develop new control methods for COMAN. As an example, a reinforcement learning algorithm was applied to tune the parameters of the linear ground model. The simulator is available for download from the IIT website [1].

As a future work, this simulator will be used to test and verify future walking controllers for uneven terrains. In addition, further optimizations will be carried out to tune the model parameters.

APPENDIX

The detailed diagram of how Robotran modules are related is illustrated in the Fig. 10. Detailed description and documentation about Robotran package is available at [22].

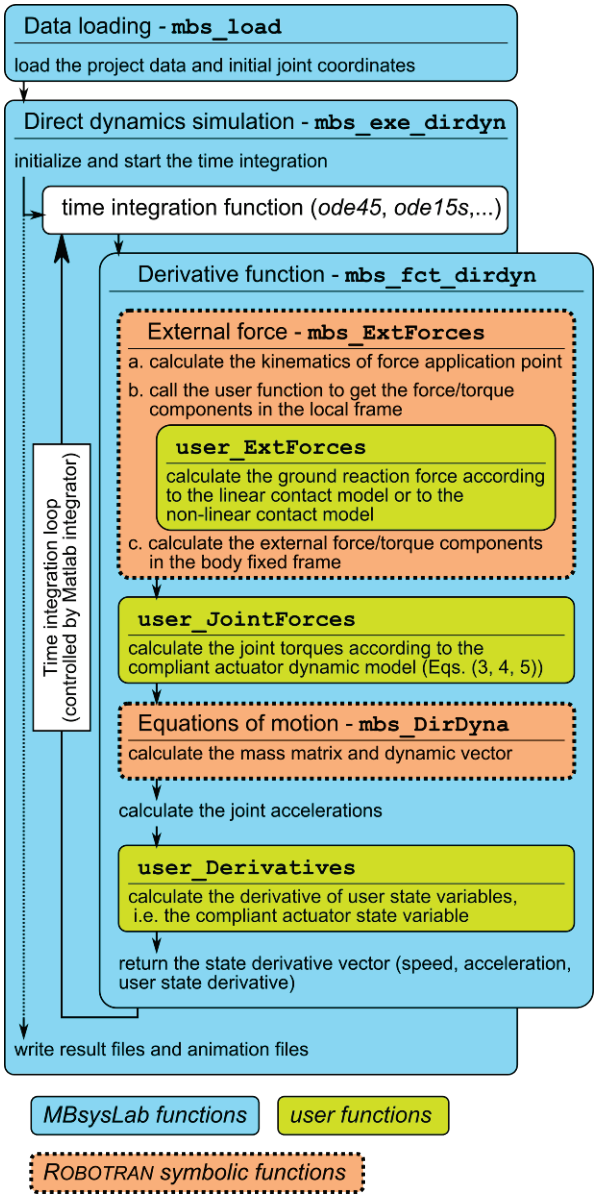

Fig. 10. Function diagram of the direct dynamics simulation environment of MBsysLab.

ACKNOWLEDGMENT

This work is supported by the European Commission FP7, "AMARSI" Project ICT-2009-4. The authors would like to thank Prof. Paul Fisette for his support.

REFERENCES

[1] (2012) Instituto Italiano di Technologia, COmpliant HuMANoid Platform (COMAN), http://www.iit.it/en/advr-labs/humanoids-a-human-centred-mechatronics/advr-humanoids-projects/compliant-humanoid-platform-coman.html.

[2] R. Smith. (2012), Open Dynamics Engine (ODE), http://www.ode.org/.

[3] O. Michel, "Webots: Professional Mobile Robot Simulation," Int. J. of Advanced Robotic Systems, vol. 1, pp. 39-42, 2004.

[4] F. Kanehiro, H. Hirukawa, and S. Kajita, "OpenHRP: Open Architecture Humanoid Robotics Platform," The Int. J. of Robotics Research, vol. 23, pp. 155-165, February 2004.

[5] S. Schaal, "The SL simulation and real-time control software package," University of Southern California, 2001.

[6] (2012) Adams Multibody Dynamics Simulation Software. http://www.mscsoftware.com/Products/CAE-Tools/Adams.aspx

[7] (2012), RoboWorks http://www.newtonium.com/index.html.

[8] (2012), MapleSim: high-performance physical modelling and simulation software. http://www.maplesoft.com/products/maplesim/.

[9] (2012), simulation & active interfaces (SAI) http://ai.stanford.edu/~conti/sai.html.

[10] J. C. Samin and P. Fisette, Symbolic Modelling of Multibody Systems, Solid Mechanics and its Applications, 1st ed.: Kluwer Academic Publishers, 2003.

[11] G. A. Medrano-Cerda, H. Dallali, M. Brown, N. G. Tsagarakis, and D. G. Caldwell, "Modelling and Simulation of the Locomotion of Humanoid Robots," in the UK Automatic Control Conference, Coventry, UK, 2010.

[12] AMARSI, Adaptive Modular Architectures for Rich Motor Skills, http://www.amarsi-project.eu/.

[13] N. G. Tsagarakis, Z. Li, J. Saglia, and D. G. Caldwell, "The design of the lower body of the compliant humanoid robot cCub," in IEEE Int. Conf. on Robotics & Automation, Shanghai, China, 2011, pp. 2035-2040.

[14] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and S. Schaal, "Compliant Quadruped Locomotion Over Rough Terrain," in IEEE/RSJ Int. Conf. on Intelligent Robots & Systems, St. Louis, USA, 2009, pp. 814-820.

[15] R. Featherstone, Rigid Body Dynamics Algorithms 1st ed. New York: Springer Science+Business Media, LLC, 2008.

[16] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in IEEE-RAS Int. Conf. on Humanoid Robots, Daejeon, Korea, 2008, pp. 22-27.

[17] M. Azad and R. Featherstone, "Modeling the contact between a rolling sphere and a compliant ground plane," in ACRA, Brisbane, Australia, 2010.

[18] S. Nakaoka, S. Hattori, F. Kanehiro, S. Kajita, and H. Hirukawa, "Constraint-based dynamics simulator for humanoid robots with shock absorbing mechanisms," in IEEE/RSJ Int. Conf. on Intelligent Robots & Systems, IROS, San Diego, CA, USA, 2007, pp. 3641-3647.

[19] P. Kormushev and D. G. Caldwell, "Simultaneous Discovery of Multiple Alternative Optimal Policies by Reinforcement Learning," in the IEEE Int. Conf. on Intelligent Systems, Sofia, Bulgaria, 2012, pp. 202-207.

[20] H. Dallali, P. Kormushev, Z. Li, and D. Caldwell, "On Global Optimization of Walking Gaits for the Compliant Humanoid Robot, COMAN Using Reinforcement Learning," Int. J. of Cybernetics & Information Technologies, vol. 12, no. 3, pp. 39-52, 2012.

[21] COMAN Simulation Accompanying Video. http://goo.gl/G6s8X .

[22] (2012) Robotran Homepage. http://www.robotran.be/.