The 6th Workshop on Humanoid Soccer Robots @ Humanoids Conference, Bled (Slovenia), October 2011.

Adaptive Dynamic Simulation Framework for Humanoid Robots

Manokhatiphaisan S. and Maneewarn T.

Abstract—**This research proposes the dynamic simulation system framework with a robot-in-the-loop concept. The proposed simulation system can communicate with the physical robot's so that the simulation model can be updated using the feedback from the robot sensor readings throughout the system development process. When the robot's hardware is modified, the robot model in the simulation can be adapted automatically to keep up with the current state of the robot. The linear inverted pendulum model (LIPM) and the three-mass inverted pendulum model (3MIPM) were used as the simplified dynamics model for a humanoid robot in the simulation system. The suggested model approximation methods including the geometry-based method and the ANN adaptation method were tested and validated in a series of experiments.**

I. Introduction

In recent decades, many research groups have tried to replicate human motions such as walking and running on humanoid robots, for instance, ASIMO [1] by Honda, HOAP [2] by Fujitsu. A humanoid robot system usually consists of many actuators which are susceptible to malfunction during long period of operation. Therefore, computer simulation software plays an important role in the development process of a humanoid robot system. In the World RoboCup humanoid soccer competition, the success of the team highly depends on the ability to perform variety of motions with sufficient degree of robustness and reliability. Some teams have implemented the robot simulation software in order to test a simple algorithm without using the real robot which can reduce the risk of the robot being damaged before the competition. However, a visual robot simulation is not sufficient to portray the robot's dynamic behavior. A physical based simulation is more effective for testing the robot. The concept of 'software in the loop' was proposed by Friedmann [3, 4] to simulate the robot under the real environmental condition. His group has proposed a multi-robot simulation system called Multi-Robot Simulation Framework (MuRoSimF) with many features including complex motion simulation and collision detection for soccer playing humanoid robots. The MuRoSimF consists of an internal sensor simulation, a camera simulation and a motion simulation. In humanoid robot development, the stability of the robot's motion is one of the major concerns. Since the humanoid robots generally have high number of degrees of freedom (DOF), its full body dynamic model is difficult to obtain and simulate. The full body dynamic simulation also requires large computational time. It is thus difficult to be used in real-time control application. A lot of research [5-8] has been done to reduce the complexity of the humanoid robot system modeling. Feng and Sun [9] have proposed the concept of a three mass linear inverted pendulum (3MLIPM) for modeling a humanoid robot. This model considers the dynamics effect of both legs rather than only the hip mass as in the linear inverted pendulum model (LIPM). Takenaka and Matsumoto [10] have proposed a method to minimize the dynamics error using ZMP trajectories compensation on a simple inverted pendulum and flywheel model. Moreover, there are also a lot of researches, for example [11-13], which uses the adaptive approach to update their models.

In this paper, the simulation software for the FIBO-KM humanoid robot (kid-size) as shown in Fig.1 is described. The concept of 'robot-in-the-loop' is introduced where the robot can send status feedback to the simulation software so that the dynamic model in the simulation software can be adapted according to the actual robot status. The LIPM and 3MLIPM are implemented in the simulation to represent the simplified dynamics model of the robot. In the 3MLIMP model, the set of robot parameters is reduced from 20 x N to 3 x N where N is the number of dynamics parameters of interest including the center of mass (CoM) and the joint torque.

This paper is organized as follows; section II describes the simulation framework. Section III describes the robot-in-the-loop concept. The simplified dynamics model is discussed in section IV. The model approximation and validation is discussed in section V and the conclusion is given in section VI.

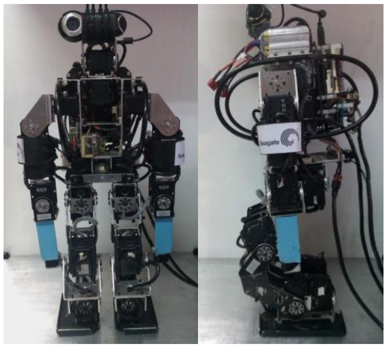


Fig. 1. FIBO-KM Humanoid Robot (Kid-Size)

II. Simulation Framework

Our simulation software was developed using several open source programming toolkits including PyODE^1, vPython^2 and numPy^3. A virtual robot is modeled as a set of virtual objects which can be either cylinder, box or sphere. The complex object can be constructed from these simple primitives. For example, the virtual motor is a combination of three boxes with different size and two cylinders. The dynamical properties such as mass, density and inertia tensor are added to the virtual object. The connection between two objects can be either a hinge joint or a fixed joint. If a virtual object is inherited from the virtual motor class, the connection between links in that object will be a virtual hinge joint as in the virtual motor class. Each virtual servo motor is controlled by the P-control algorithm. The class diagram of the virtual robot model is shown in Fig. 2. In order to develop this simulation software to be used for the RoboCup humanoid soccer competition, the RoboCup soccer field is simulated. In the virtual world, the gravitational force is set to 9.81m/s^2. In the simulation software, all services such as an input event and a visual update are managed and handles in the main control loop. The captured screen of the simulation software is shown in Fig. 3. Each virtual servo motor is controlled by the P-control algorithm.

III. Robot-In-The-Loop Framework

In most robot simulation system, the robot parameters (such as mass matrix of each link, joint limits, etc.) are assigned at the beginning phase of the simulation system development. The simulation system is usually used separately from the actual robot. When the robot is mechanically modified for the test or the competition, the simulation is not necessarily updated. Thus, the simulation can sometimes misrepresent the current state of the actual robot. We propose the robot-in-the-loop framework where we design our simulation system so that it can easily connect with the real robot such that the robot model in the simulation can be updated from the feedback of the real robot sensor readings throughout the system development process. When the robot's hardware is modified, the robot model in the simulation can be adapted automatically to represent the current state of the robot. The FIBO Humanoid Robot KM-Series (Kid Size) has 20- DOFs; 12-DOFs on legs, 6-DOFs on arms and 2-DOFs on head. The commercial digital servo motors (robotis-Dynamixel) were used (18 of RX- 28 and 2 of RX-64). The robot height is about 53 cm and its weight is about 3.3 kg. The electronics system in the robot can be divided to four sections; main computer, motion controller, internal sensors and servo motor. In the real robot, the PICO iTX 820, an INTEL ATOM® processor at 1.66GHz with 1GB of RAM, is used as the main computer. The micro-controller ARM7 LPC2148 is the motion controller. The internal sensors consist of a 3-DOF accelerometer and a 2-DOF gyroscope. The power source of the robot is 2 sets of 4-cells Li-Po batteries. One set supports the main computer. Another set supports all motors and the main controller. The simulation program is running on a MacBook Pro with 3 GB of RAM. The communication diagram between the real robot and the simulation software is shown in Fig. 4.

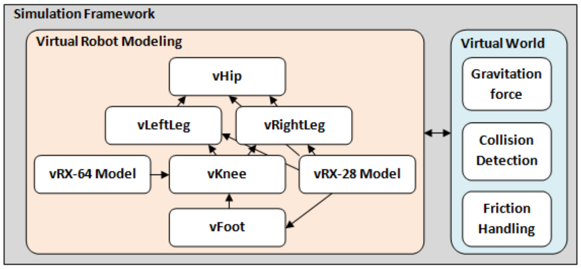
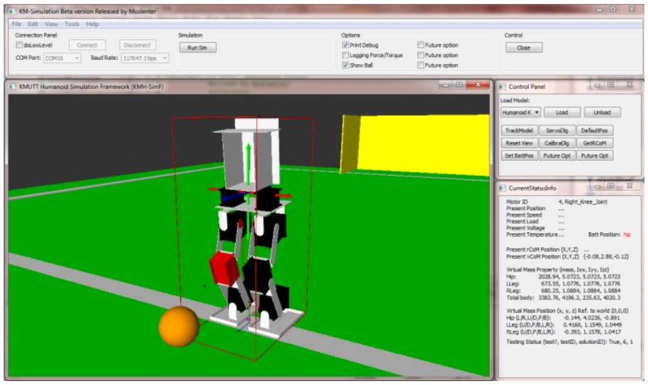


Fig. 2. Virtual robot class diagram



□ doLowLevel

COM Port: COM16

Baud Rate: 117647.1 bps

☑ Print Debug

□ Logging Force/Torque

☑ Show Ball

□ Future option

□ Future option

□ Future option

Humanoid K

```
Motor ID                 4, Right_Knee_Joint
Present Position         ...
Present Speed            ...
Present Load             ...
Present Voltage          ...
Present Temperature...             Batt Position: hip

Present rCoM Position (X,Y,Z)   ...
Present vCoM Position (X,Y,Z)   (-0.08,2.88,-0.12)

Virtual Mass Property (mass, Ixx, Iyy, Izz)
Hip:          2028.94, 5.0723, 5.0723, 5.0723
LLeg:          673.55, 1.0776, 1.0776, 1.0776
RLeg:          680.25, 1.0884, 1.0884, 1.0884
Total body:   3382.76, 4196.2, 235.63, 4020.3

Virtual Mass Position (x, y, z) Ref. to world (0,0,0)
Hip (L/R,U/D,F/B):      -0.144, 4.0236, -0.891
LLeg (U/D,F/B,L/R):     0.4168, 1.1549, 1.0449
RLeg (U/D,F/B,L/R):     -0.393, 1.1578, 1.0417
Testing Status (test?, testID, solutionID): True, 6, 1
```

Fig. 3. KMUTT Humanoid Simulation Framework (KMH-SimF)

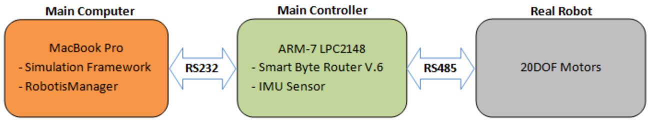


Fig. 4. Simulation Communication Architecture

IV. Simplified Dynamics Model

A. Linear Inverted Pendulum Model (LIPM)

It is difficult to model the humanoid robot exactly with all its joints and links in order to perform the dynamic motion control in real time. Thus, a simplified dynamics model is used instead [5-10]. The Linear Inverted Pendulum Model (LIPM) is generally used to represent the dynamics of the humanoid robot in the simplest form. The equation of motion with the LIPM [10] in the Sagittal plane can be shown as follows:

$$m_{pend} l\ddot{x}_{pend} = m_{pend}(g+\ddot{z}_{pend})*x_{pend}+M_{pend} \tag{1}$$

$$F_{pend} = m_{pend}\ddot{x}_{pend} \tag{2}$$

$$\Delta M_{pend} = l\Delta F_{pend} \tag{3}$$

[1] http://pyode.sourceforge.net

[2] http://vpython.org

[3] http://numpy.scipy.org

where, x_{pend} is the horizontal position of the pendulum, m_{pend} is the mass of the pendulum. F_{pend} is the ground reaction force of the pendulum, M_{pend} is the ground reaction moment of the pendulum, $\ddot{z}_{pend}$ is the vertical acceleration of the pendulum, l is the length of the pendulum and g is the gravitational acceleration constant.

B. Three Mass Linear Inverted Pendulum Model (3MLIPM)

The 3MLIPM [9] includes the effect from both legs of the robot in addition to its body mass which makes this model representation more precise compared to the LIPM. In this model, the robot can be divided into three parts which are the left leg, the right leg and the hip. Each part is represented by a point mass with a mass-less link. The illustration of this model is shown in Fig. 5. The equation of motion at the support ankle is:

$$\tau_z = \sum_{i=1}^{3} m_i \left(\ddot{x}_1 + g\right) y_i - \sum_{i=1}^{3} m_i \ddot{y}_i x_i \tag{4}$$

$$\tau_y = \sum_{i=1}^{3} m_i \ddot{z}_i x_i - \sum_{i=1}^{3} m_i \left(\ddot{x}_i + g\right) z_i \tag{5}$$

where, (x_i, y_i, z_i) is the position of m_i , and $(0,\tau_y, \tau_z)$ is the supporting ankle torque.

In LIPM, the model is defined under the assumption that the pendulum length is kept constant and the supporting ankle torque is zero while the robot is walking. These two assumptions are also taken in the 3MLIPM. These assumptions allow equation (4) and (5) to be simplified to equation (6) and (7) respectively. The robot motion in Sagittal and Frontal plane can be generated separately.

$$\sum_{i=1}^{3} m_i \ddot{y}_i x_i = \sum_{i=1}^{3} m_i g y_i \tag{6}$$

$$\sum_{i=1}^{3} m_i \ddot{z}_i x_i = \sum_{i=1}^{3} m_i g z_i \tag{7}$$

For our FIBO-KM humanoid robot, the mass property of the robot is shown in Table I. Both arms, a head and a body will be included in the hip mass.

TABLE I
MASS PROPERTY OF THE ROBOT

Position	Inertia Tensor [Ixx,Iyy,Izz] in kg*m^2	Mass (kg)	% of total mass
Right leg	[0.00279, 0.003383, 0.001256]	0.743	21.62
Left leg	[0.00343, 0.053453, 0.047799]	0.740	21.53
Torso	[0.00352, 0.006423, 0.004844]	1.375	42.92
Right arm	[0.00060, 0.000125, 0.000628]	0.238	6.92
Left arm	[0.00066, 0.000254, 0.000502]	0.240	6.98

*All inertia tensor values are extracted from SolidWorks

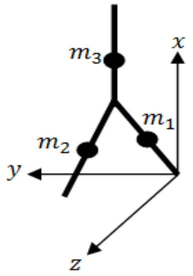


Fig. 5. The illustration of 3MLIPM and reference plane

V. MODEL APPROXIMATION & VALIDATION

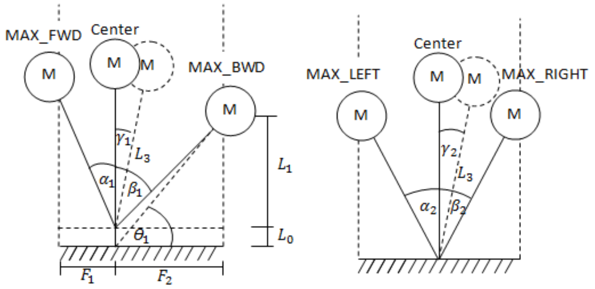
In this section, the experiments for validating the simulated model in LIPM, 3MLIPM will be discussed. The proposed method for an automatic model adaptation using artificial neural networks (ANN) is also explained and discussed.

Some assumptions must be introduced as follows.

- The ground is flat.
- The foot position is always in the same point.
- No slip on the ground.
- The robot body is rigid.
- There is a constant voltage from the switching power supply.

A. LIPM Approximation

In the LIPM, a humanoid robot is modeled to a simple mass on the fixed length pendulum. Therefore, the position of the robot's center of mass (CoM) in Cartesian space can be estimated based on the known falling angles (backward/forward) of its ankle. The illustration of the geometry based CoM approximation is shown in Fig. 6.



(a) (b)

Fig. 6. An illustration of geometry based CoM approximation

With this model, the CoM of the single mass can be computed geometrically from the angle readings of the ankle joint. In this calculation, some constants must be defined. F_1 = 3.5 cm., F_2 = 9.5 cm., L_0 = 3.1 cm. and φ = 2.2 where, F_1, F_2, L_0 and φ are the distance from the ankle to the front of the foot plate, the distance from the ankle to the back of the foot plate, the offset between the foot plate and the ankle rotation point, and the single support ratio respectively. The result of approximation CoM in (x,y,z) is (L_3,W_1,D_1). According to the Robotis potentiometer sensor in the servo motor, the degree is determined by the following equations:

$$\begin{aligned} &\alpha_1=(Center - Max_{Fwd})*300/1024 \\ &\beta_1= (Max_{Bwd} -Center)*300/1024 \\ &\alpha_2=(Center – Max_{Left})*300/1024 \\ &\beta_2= (Max_{Right} -Center)*300/1024 \end{aligned} \tag{8}$$

Where α_1, β_1, α_2 and β_2 are max forward degree, max backward degree, max left degree, and max right degree respectively. The Z-axis degree (γ_1) equation is:

$$\gamma_1 = \frac{\alpha_1+\beta_1}{2} - \alpha_1 \tag{9}$$

The pendulum length is calculated as follows:

$$L_1 = \tan (90- \beta_1) \times F_2 \tag{10}$$

$$L_3 = \frac{F_2}{\cos \theta_1}, where\ \theta_1 = \tan^{-1}\left(\frac{L_1 + L_0}{F_2}\right) \tag{11}$$

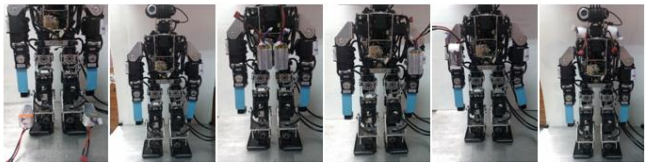
The equation about Z-axis is follows:

$$D_1 = L_3 \times \tan (\gamma_1) \tag{12}$$

In Y-axis the equation is shown as follows:

$$\begin{aligned} \gamma_2 &= \begin{cases} \alpha_2 - \beta_2, \alpha_2 \geq \beta_2 \\ \beta_2 - \alpha_2, \alpha_2 < \beta_2 \end{cases} \\ W_1 &= \begin{cases} -L_3 \times \tan(\gamma_2), \alpha_2 \geq \beta_2 \\ L_3 \times \tan(\gamma_2), \alpha_2 < \beta_2 \end{cases} \end{aligned} \tag{13}$$

In order to verify the approximated CoM by the geometry-based method, the result of the CoM from the geometry-based method were compared with the physical measurement (i.e. the classical approach) by using mass-less plate and the straight rod. In the classical approach, the measurement is performed when the robot is in the standing upright position (with fully extended knees). By laying the robot down on the mass-less plate, the approximate height of CoM can be found. Five testing scenarios were tested. In each scenario, the battery was mounted at different locations on the robot in order to represent a variation in mass location. In scenario a to f, the battery is mounted at both legs, no battery, the battery is mounted at left arm, right arm and shoulders respectively. The illustration of testing scenarios is shown in Fig. 7. The estimated CoM of the robot from the two approaches (i.e. the classical and the geometrical-base CoM approximation) in six scenarios is shown in Table II.



Leg (a) None (b) Front Hip (c) Left Arm (d) Right Arm (e) Shoulder (f)

Fig. 7. Testing scenarios

TABLE II
COM TESTING RESULT

No.	Classical Approach(cm)	Geometry Based (cm)	Abs Error in X-axis (cm)	Abs Error in Y-axis (cm)	Abs Error in Z-axis (cm)
a	20.0,0.5,-1.7	21.0,0.6,-2.2	1.0	0.1	0.5
b	22.5,0.5,-1.3	25.1,0.9,-1.8	2.6	0.4	0.5
c	22.3,0.0.-1.7	23.8,0.0,-2.3	1.5	0.1	0.6
d	24.5,-0.3,-1.5	26.5,-0.1,-2.1	2.0	0.2	0.6
e	24.0,1.7,-1.5	26.0,2.0,-2.1	2.0	0.3	0.6
f	25.7,0.7,-1.2	27.6,1.0,-1.9	1.9	0.3	0.7

From the result in Table II, the error is calculated based on the estimated CoM from the classical approach.

TABLE III
TESTING SCENARIO AND RESULT

No.	Batteries Position	Mass (m_1,m_2,m_3) in Kg.	CoM (x,y,z) position(cm.)
a	Leg	(0.840, 0.843, 1.82)	21.7,0.8,0.3
b	-	(0.740, 0.743, 1.82)	25.3,0.8,1.6
c	Front Hip	(0.740, 0.743, 2.02)	26.9,0.5,1.6
d	Left Arm	(0.740, 0.743, 2.02)	23.9,-0.5,0.6
e	Right Arm	(0.740, 0.743, 2.02)	25.8,1.9,1.2
f	Shoulder	(0.740, 0.743, 2.02)	25.8,1.0,1.7

-In testing No.b, there is no battery added in the robot
-The CoM value is from the average value from the ten trials solution.

TABLE IV
SETS OF TESTING PARAMETERS

No.	Falling Forward Degree(°)		Falling Backward Degree(°)		Time to Fall Forward (s)	Time to Fall Backward (s)
	Send Pos	Real Pos	Send Pos.	Real Pos.		
a	7.61	8.78	19.92	24.31	5.56	14.64
b	8.49	8.78	16.69	19.04	6.33	11.67
c	6.73	7.61	17.57	18.46	5.51	11.32
d	7.32	9.38	15.72	20.50	5.56	11.72
e	7.03	7.91	16.11	19.33	5.13	11.84
f	7.3	8.49	15.23	18.75	5.56	10.95

B. 3MLIPM Approximation

In the second experiment, the objective is to verify the modeling accuracy of the virtual robot in the simulation framework with the LIPM and 3MLIPM concept in comparison with the real robot. First the robot was commanded to move forward/backward in Sagittal plane until the falling angles can be measured and the CoM can be geometrically approximated as described in the previous experiment. Five scenarios as explained in the previous experiment were tested.

In the LIPM concept, the virtual robot was modeled as a single mass with the position of the mass at the approximated CoM position showed in table II. In the 3MLIPM concept, the three masses are assigned to m_1, m_2 and m_3 as shown in the third column of Table III. The position of m_1, m_2 and m_3 in Cartesian coordinate which comprises of 9 parameters in total, are chosen manually. The chosen position of these three masses still keep the CoM of the virtual robot (as shown in the fourth column of Table III) to be reasonably close to the CoM position in the LIPM case in table II.

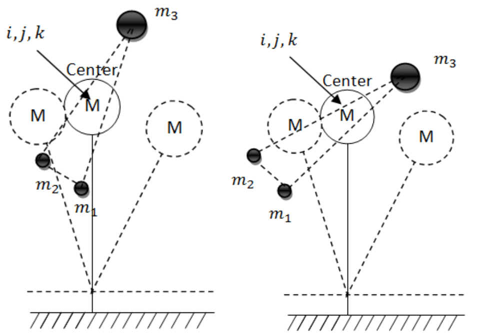


Fig 8. The illustration of comparison testing between models

TABLE V
COMPARISON BETWEEN MODEL RESULTS

No.	Sending Angles (°)		3MLIPM Falling Angles (°)		LIPM Falling Angles (°)		Abs Angles Error in LIPM (°)	
	FW	BW	FW	BW	FW	BW	FW	BW
a	8.78	24.31	8.78	24.31	8.78	25.21	0	0.9
b	8.78	19.04	8.78	19.04	9.39	19.04	0.61	0
c	7.61	18.46	7.61	18.46	7.61	20.78	0	2.32
d	9.38	20.50	9.38	20.50	9.59	20.50	0.21	0
e	7.91	19.33	7.91	19.33	8.58	19.33	0.67	0
f	8.49	18.75	8.49	18.75	8.68	18.75	0.19	0

We assume that if the simplified model of the virtual robot can successfully represent the real robot, the behavior of falling forward and backward in the simulation should be the same as in the real robot. If the real robot falls forward at the falling angle θ_{fall_fwd} at time t_1 and falls backward at the falling angle θ_{fall_bwd} at time t_2, the virtual robot must be falling at the same time and same falling angle in both forward and backward cases. In each test, the commands were sent to the robot to increase or decrease its ankle angular position from zero standing pose until it fell forward or backward. In Table IV, the send position means the angular position command that was sent to the robot when it fell. The real position means the angular position readings that was read from the robot ankle when it fallen. Time to fall forward and time to fall backward are time were measured from the robot's initial zero pose until the robot fell. Table IV, shows the sets of testing parameters that were sent to and read from the robot when it fell in six different test cases. Table V compares the behavior of the LIPM vs. the 3MLIPM. The second column shows the angular position commands that were sent to the real robot when it fell. The falling angles in the third and the fourth column are the ankle's angle measurement of the virtual robot in the simulation when it fell. Even when the virtual CoM in the 3MLIPM model is very close to the LIPM, the falling angles of the simulation with the LIPM were, in some cases, different from the actual robot. The virtual robot with the LIPM can satisfy only forward or backward falling. The 3MLIPM can better represent the falling behavior of the actual robot as shown by the same falling angles in all six test scenarios. Fig. 8 illustrated cases of different three masses position that resulted in the same virtual CoM (shown at (i,j,k)) for the single mass model. However, the falling angle in these two cases can be significantly different.

TABLE VI
NN TESTING RESULT

No.	Sending Angles (°)		M_p	Manual vCoM (x,y,z)	NN vCoM (x,y,z)	Falling Angles (°)	
	FW	BW				FW	BW
a	8.78	24.31	0	21.7,0.8,0.3	24.7,0.7,0.7	8.78	24.31
b	8.78	19.04	1	25.3,0.8,1.6	25.8,0.8,1.7	8.78	19.04
c	7.61	18.46	2	26.9,0.5,1.6	27.0,0.8,1.7	7.61	18.46
d	9.38	20.50	2	23.9,-0.5,0.6	24.1,-0.5,0.8	9.38	20.50
e	7.91	19.33	2	25.8,1.9,1.2	26.0,2.0,1.5	7.91	19.33
f	8.49	18.75	2	25.8,1.0,1.7	26.2,1.3,1.6	8.49	18.75

TABLE VII
NN VALIDATING RESULT

No.	Sending Angles (°)		M_p	NN vCoM (x,y,z)	Falling Angles (°)	
	FW	BW			FW	BW
a	8.80	24.29	0	23.7,0.7,0.7	8.80	24.29
b	8.76	19.02	1	25.8,0.8,1.7	8.76	19.02
c	7.59	18.48	2	27.2,0.9,1.7	7.59	18.48
d	9.40	20.48	2	24.2,-0.5,0.9	9.40	20.48
e	7.95	19.29	2	26.2,2.0,1.2	7.95	19.29
f	8.45	18.79	2	26.8,1.8,1.8	8.45	18.79

C. ANN-Based Model Adaptation

In the second experiment, the three masses position is chosen by human according to the approximated CoM that is geometrically calculated from the robot's falling angles. In order to make this model adaptation mechanism autonomous, the learning algorithm was applied. For this model adaptation mechanism, the inputs as observed from the previous process are the forward/backward falling angle and the general location of the changing mass. The outputs are the location of the three masses in Cartesian coordinate. The artificial neural networks (ANN) was chosen as a tool for learning the human skill involved in this particular problem. From the experiment in section B, the input of the system are the forward falling angle θ_{fall_fwd}, the backward falling angle θ_{fall_bwd} and the general location of an additional mass (i.e. battery in this experiment) m_p. The position *(x,y,z)* of each point mass is the output of the system. Input and output data set from the previous experiment are divided into three groups: training set, testing set and validated set. Using the 3 layered perceptrons with 5, 10 and 5 neurons for layer 1, 2 and 3 respectively with back-propagation learning, the position of the three masses can be generated from the three inputs (forward/backward falling angles and general battery position). Table VI shows the virtual CoM computed from the three masses position generated by the ANN and the manual method using the testing data set. The falling angles of the virtual robot that modeled by the ANN resulted in the same falling angles with the real robot. In the validate case shown in Table VII, the model generated from ANN also correctly represented the falling angles of the real robot.

VI. CONCLUSION AND FUTURE WORK

In this paper, the simplified dynamics model was implemented in our proposed robot-in-the-loop simulation framework for a humanoid robot. Two simplified dynamics model: the LIPM and the 3MLIPM were applied. The proposed simulation framework has the ability to connect to the real robot and obtain physical information from the robot such as joint angles, joint torque in real time. By commanding the robot to roll its ankle forward and backward, the ankle's angle when the robot is falling can be recorded and used. Base on this information, the geometry-based CoM approximation method can be applied. When the robot is modified by either adding or moving some components, the robot can easily be modeled in the LIPM using the geometry-based CoM approximation approach. Moreover, the 3MLIPM can also be constructed manually based on the approximated CoM and the general location of the added component. The accuracy of the LIPM and the 3MLIPM virtual robot was compared. The 3MLIPM showed more accurate falling behavior of the real robot than the LIPM. The virtual robot in the proposed simulation framework can successfully represent the dynamic behavior of the real robot. Additionally, the automatic model adaptation mechanism was also implemented. The ANN is used to learn human skills in generating the three masses location from the falling angles and the general location of the added mass. The 3MLIPM can be automatically generated correctly by the ANN. With the proposed simulation framework, the virtual robot model can be generated and adapted when the robot is physically modified. The simplified dynamics approach is much easier to implement and less complex compared to the full body dynamics model, thus required less parameters to adapt when the real robot is changed or modified. In the near future, the proposed simulation framework will be used for designing and testing walking gaits and other motions of our humanoid robot.

ACKNOWLEDGMENT

This research work is financially supported by the National Science and Technology Development Agency (NSTDA) and Institute of Field Robotics (FIBO), King Mongkut's University of Technology Thonburi.

REFERENCES

[1] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent ASIMO: system overview and integration," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, 2002, pp. 2478-2483 vol.3.

[2] A. Weiss, J. Igelsbock, S. Calinon, A. Billard, and M. Tscheligi, "Teaching a humanoid: A user study on learning by demonstration with HOAP-3," in *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, 2009, pp. 147-152.

[3] M. Friedmann, K. Petersen, and O. von Stryk, "Simulation of Multi-Robot Teams with Flexible Level of Detail," in *Proceedings of the 1st International Conference on Simulation, Modeling, and Programming for Autonomous Robots* Venice, Italy: Springer-Verlag, 2008.

[4] M. Friedmann, K. Petersen, and O. von Stryk, "Adequate motion simulation and collision detection for soccer playing humanoid robots," *Robotics and Autonomous Systems,* vol. 57, pp. 786-795, 2009.

[5] N. Motoi, T. Suzuki, and K. Ohnishi, "A Bipedal Locomotion Planning Based on Virtual Linear Inverted Pendulum Mode," *Industrial Electronics, IEEE Transactions on,* vol. 56, pp. 54-61, 2009.

[6] W. Wu, Y. Wang, Y. Pan, and F. Liang, "Research on the Walking Modes Shifting Based on the Variable ZMP and 3-D.O.F Inverted Pendulum Model for a Humanoid and Gorilla Robot," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 1978-1983.

[7] K. Erbatur and U. Seven, "An inverted pendulum based approach to biped trajectory generation with swing leg dynamics," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, 2007, pp. 216-221.

[8] F. Shuai and S. Zengqi, "A simple trajectory generation method for biped walking," in *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, 2008, pp. 2078-2082.

[9] C. Xiong, H. Liu, Y. Huang, Y. Xiong, S. Feng, and Z. Sun, "Biped Robot Walking Using Three-Mass Linear Inverted Pendulum Model," in *Intelligent Robotics and Applications*. vol. 5314: Springer Berlin / Heidelberg, 2008, pp. 371-380.

[10] T. Takenaka, T. Matsumoto, and T. Yoshiike, "Real time motion generation and control for biped robot -3rd report: Dynamics error compensation," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 1594-1600.

[11] K. Sujeong, R. Stephane, and J. K. Young, "Continuous collision detection for adaptive simulation of articulated bodies," *Vis. Comput.,* vol. 24, pp. 261-269, 2008.

[12] S. T. Hansen, M. Svenstrup, and L. Dalgaard, "An Adaptive Robot Game," *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK),* pp. 1-8.

[13] A. Miller, P. Allen, V. Santos, and F. Valero-Cuevas, "From Robot Hands to Human Hands: A Visualization and Simulation Engine for Grasping Research," *Industrial Robot,* vol. 32, pp. 55-63, 2005.