REFLEX BASED WALKING PATTERN ADAPTATION FOR BIPED ROBOTS

By

YASSER EL-KAHLOUT

Submitted to the Graduate School of Engineering and Natural Sciences in partial fulfillment of the requirements for the degree of Master of Science

SABANCI UNIVERSITY

July 2003

REFLEX BASED WALKING PATTERN ADAPTATION FOR BIPED ROBOTS

APPROVED BY:

Assist. Prof. Dr. Kemalettin Erbatur ……
(Thesis Advisor)

Assist. Prof. Dr. Gökhan Göktuğ ……
(Thesis Co-Advisor)

Prof. Dr. Asif Şabanoviç ……

Assist. Prof. Dr. Erhan Budak ……

Assist. Prof. Dr. Ahmet Onat ……

DATE OF APPROVAL ……

ABSTRACT

Employing robots to replace humans in heavy and dangerous tasks is an important research area. Biped robots have advantages in obstacle avoidance and are therefore suitable to work in the human environment in such tasks. However, their control is a very difficult problem because of their nonlinear and unstable nature. Even very small disturbances can lead to instability. Disturbances can vary from slippery ground surfaces to collisions and unexpected contact with the environment to variations in the payload.

For dynamically stable robots (walking on two or less feet), constraints on timing and foot placement increase the difficulty of designing controllers that can anticipate changes in the payload or react to errors. This thesis demonstrates the effectiveness of preprogrammed high-level responses to locomotion in a complex dynamic environment. A suite of responses allows a simulated, three dimensional, bipedal robot to recover from falling down due to a sudden change in the payload.

Many environment contact errors would be avoided if the control system can respond fast to the errors that have already taken place and adapt the biped locomotion. In the case of the biped robot considered in this work, the controller might have less than a few tenths of a second in which to choose or plan an appropriate recovery. In this thesis reflexes are defined as responses with no explicit modeling and limited sensing. That is the robot can detect the payload change and makes no attempt to estimate the properties of the load to calculate a corresponding recovery plan. These reflexes are defined at high level because they involve changes of the biped body configuration and trajectory. Sensing elements are used just to detect the error and trigger the reflex.

Explicit dynamic modeling of the biped robot is complicated and the controller cannot use it to compute precise and appropriate reactions. In addition, accurate and precise information on load addition is not available to the controller. The method presented changes the walk trajectory and shifts the center of gravity to keep the balance of the walk.

Thereafter, the original trajectory is brought back by a smooth trajectory interpolation function. The reflex-adaptation technique considered is tested for a variety of payloads at different loading times.

The method shows a good functionality by recovering the biped and allowing stable and balanced original walking pattern. The approach is successful and is a candidate for real applications.

ÖZET

Ağır ve tehlikeli işlerde, insanların yerine robotların görevlendirilmesi önemli bir araştırma alanıdır. Yürüyen robotlar, engellerden kurtulmada bir çok avantaja sahiptir ve insan ortamında kullanılmak için uygundurlar. Ancak, yürüyen robotların denetimi doğrusal olmayan ve değişken özellikleri yüzünden çözülmesi zor bir problemdir. Küçük bir ortam değişikliği bile dengenin yitirilmesine sebep olabilir. Değişiklikler, kaygan zemin yüzeylerinden çarpışmalara, ortamla beklenmeyen temaslardan yükteki değişimlere kadar çeşitlilik gösterirler.

İki veya daha az ayak ile yürüyen hareketli robotlar için zamanlama ve ayak basma kısıtlamaları, engebeli araziyi önceden tespit edebilen veya hatalara karşı tepki gösterebilen denetimci tasarlanmasındaki zorlukları arttırmaktadır. Bu tezde karmaşık hareketli ortamda önceden programlanmış yüksek seviyeli tepkilerin etkinliği gösterilmektedir. Bir grup tepki, benzeşim ortamında yürüyen üç boyutlu robotun ani yük değişiklerine bağlı olarak düşmesini engellemeyi sağlamaktadır.

Kontrol sistemi ortaya çıkmış hatalara hızlı tepki verebilir ve yürüyen robotun hareketi uyarlanırsa, birçok ortam temas hatası önlenebilir. Tezde ele alınan yürüyen robotta, denetimci uygun düzeltmeyi seçmek ya da planlamak için saniyenin onda birinden daha az bir zamana sahiptir. Bu tezde, refleksler net bir modellemesi olmayan, kısıtlı duyumlu tepkiler olarak tanımlanmıştır. Bir diğer deyişle, robot yükteki değişimi bulabilmekte fakat uygun iyileştirme planı için yükün özelliklerini tahmin etme amaçlı hiçbir girişim yapmamaktadır. Refleksler yüksek seviyede tanımlanmaktadır ve yürüyen gövde konfigürasyonun değişim ve yörüngelerini de içermektedirler. Duyum elemanları sadece hatayı tespit etmek ve refleksi tetiklemek için kullanılmaktadır.

Denetimci planlı tepkiler hesaplamak için yeterli süreye sahip değildir. Bunlara ek olarak, yük eklenmesi ile ilgili doğru ve kesin bir bilgiden yoksundur. Anlatılan metot yürüyüşün dengesini korumak için, yürüyüş yörüngesini değiştirmekte ve ağırlık merkezini

kaydırmaktadır. Orijinal yörünge, enterpolasyon fonksiyonu ile yumuşak geçişli bir yörüngeye değiştirilir. İncelenen refleks-adaptasyon metodu farklı yükler için değişik pozisyon ve zamanlarda test edilmiştir.

Sunulan yöntem dengeli yürüyüşün devamını sağlayarak iyi bir işlev göstermektedir. Yaklaşım başarılıdır ve gerçek uygulamalar için bir adaydır.

Dedicated to my lovely wife, İlknur

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my advisor Assistant Prof. Dr. Kemalettin Erbatur, for the guidance, encouragement, and feedback he provided to me during my thesis studies.

I would like to thank my father, mother, sister, and brother for their support and encouragement. I would like to add special thanks to my wife Ilknur for her being always next to me in this time. I would like to thank my professors and my friends in the faculty for everything.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

TABLE OF SYMBOLS

X_r	Right foot x position referenced in body frame
X_l	Left foot x position referenced in body frame
Y_r	Right foot y position referenced in body frame
Y_l	Left foot y position referenced in body frame
Z_r	Right foot z position referenced in body frame
Z_l	Left foot z position referenced in body frame
t_{ss}	Single support phase period
t_{ds}	Double support phase period
b_s	Half amplitude of X_r or X_l
x_{ref_as}	x coordinate of the feet center in body frame
w_{of}	Distance between body frame and any foot frame in y direction if swing is equal to zero
w	Half amplitude of Y_r or Y_l
y_{ref_as}	y coordinate of the feet center in body frame
h_b	Body height, elevation of the body frame
h_s	Amplitude of Z_r or Z_l
x_{r_s}	Right foot x position at the beginning of the reflex
x_{l_s}	Left foot x position at the beginning of the reflex
y_{r_s}	Right foot y position at the beginning of the reflex
y_{l_s}	Left foot y position at the beginning of the reflex
z_{r_s}	Right foot z position at the beginning of the reflex
z_{l_s}	Left foot z position at the beginning of the reflex
x_{r_e}	Right foot x position at the end of the reflex
x_{l_e}	Left foot x position at the end of the reflex
y_{r_e}	Right foot y position at the end of the reflex

y_{l_e}	Left foot y position at the end of the reflex
z_{r_e}	Right foot z position at the end of the reflex
z_{l_e}	Left foot z position at the end of the reflex
x_{refl_r}	Value added to X_r during the reflex period
x_{refl_l}	Value added to X_l during the reflex period
y_{refl_r}	Value added to Y_r during the reflex period
y_{refl_l}	Value added to Y_l during the reflex period
z_{refl_r}	Value added to Z_r during the reflex period
z_{refl_l}	Value added to Z_l during the reflex period
t_{refl_per}	Time duration in which the reflex is applied
Δt	Simulation step time

LIST OF ABBREVIATIONS

ZMP	Zero Moment Point
ASV	Adaptive Suspension Vehicle
m	Meter
s	Second

CHAPTER 1

INTRODUCTION

1.1. Motivation

Employing robots to replace humans in heavy and dangerous tasks is an important research area. Biped robots have advantages in obstacle avoidance and are therefore they are suitable to work in the human environment in such tasks. However, their control is a very difficult problem because of their nonlinear and unstable nature. Even very small disturbances can lead to instability. Disturbances can vary from slippery ground surfaces to collisions and unexpected contact with the environment to variations in the payload.

Statically stable robots, which always maintain their balance over at least three legs, have used controllers with foot-placement algorithms to achieve viable footholds. However for dynamically stable robots like bipeds constraints on timing and foot placement increase the difficulty of designing controllers that can anticipate rough terrain or react to errors. This thesis demonstrates the effectiveness of preprogrammed high-level responses to locomotion in a complex dynamic environment. A suite of responses allows a simulated, three dimensional, bipedal robot to recover from falling down due to a sudden changes in the payload.

If the robot cannot detect and avoid or prepare for environment features and errors, then robust locomotion requires that the robot respond to unexpected features after the contact error has occurred and before the robot crashes. For a dynamically stable robot, the time available for modeling the environment and planning an appropriate reaction is severely limited. In the case of the biped robot considered in this thesis and shown in Figure 1.1, the controller may have less than a few tenths of a second in which to choose or

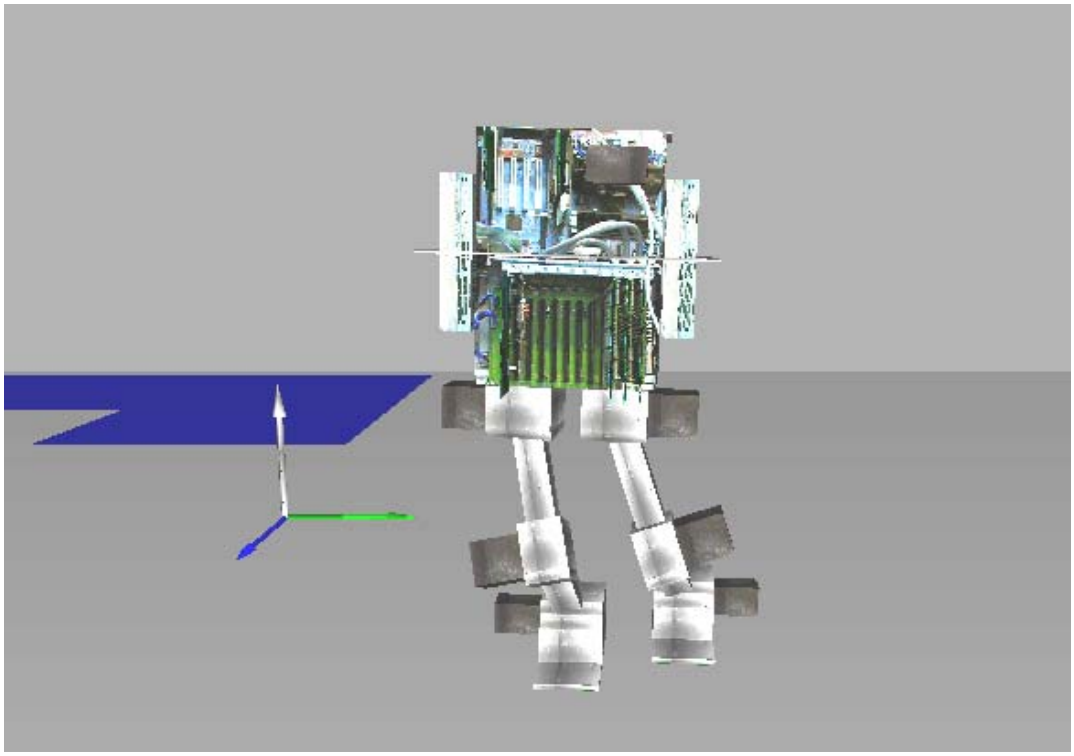

Figure 1.1. The twelve dof bipedal robot used in simulation

plan an appropriate recovery. In this work, reflexes are defined as responses with no explicit modeling and with limited sensing. That is, the robot can detect the payload change and makes no attempt to estimate the properties of the load to devise a corresponding recovery strategy. The reflexes are defined at high level because they involve changes of the biped position references. Sensing is used just to detect the error and trigger the reflex.

1.2. Statement of the Problem

This thesis proposes a simple but effective on-line gait adaptation method to recover a 12 dof dynamically modeled biped walking robot from falling down due to an abrupt addition of a load. The case studied is identified by a sudden attachment of the load to the robot at the front of its trunk. Adding the load unexpectedly, the robot may not have

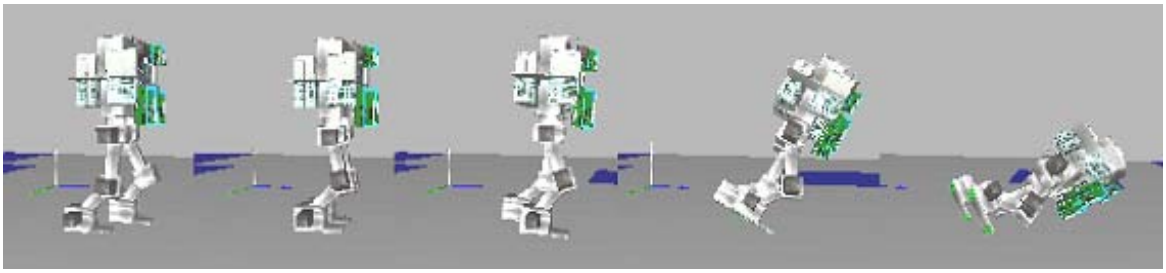

Figure 1.2. Snapshots for the robot with no reflex applied in the swing phase

sufficient time to respond to this action. Also, explicit dynamic modeling of the biped robot is complicated and the controller cannot use it to compute precise and appropriate reactions. In addition, accurate and precise information on load addition is not available to the controller. The value of mass added is an unknown parameter. The method presented changes the walk trajectory and shifts the center of gravity of the body to keep the balance of the walk. Figure 1.2 takes snapshots for the robot with a load applied during the single support phase. No reflex or adaptation technique is applied at this stage.

Chapter two surveys studies on reflex based control. Chapter three illustrates the reflex-adaptation technique. Results and modifications are displayed in chapter four. Conclusions are presented lastly in chapter five.

CHAPTER 2

A SURVEY ON REFLEX BASED CONTROL

The adaptation of the walking pattern to compensate for the disturbances and to keep the balanced biped walk is studied intensively by many researchers. Also a number of studies are reported on gait adaptation based compensation techniques on multi-legged robots. Some researchers propose a method based on visual perception to modify the walking pattern. However, in many cases disturbances or irregularities in the environment cannot be sensed before they affect the stability of the robot. The use of information from sensors might be impossible in some environments because sensor data may get contaminated or in general many sensors do not have adequate bandwidth to allow fast detection of critical conditions. Almost all approaches address the case where disturbances are only sensed after they affect the robot. In those studies, reactive motion patterns, which are artificial equivalents of reflexes of living beings, are employed for the recovery from instabilities in the robot walk.

Biological systems use many different reflexes in locomotion and manipulation. Reflexes help to restore balance and when perturbations occur during walking or standing. The role of reflexes in walking is complicated: The same stimulus elicits a different response in the stance phase (two feet are touching the ground) than in the swing phase(one foot is touching the ground) [1-3]. Touching the foot of a cat or human during a swing phase will cause the leg to flex, raising the foot. If an obstacle cause the stimulus, this response might lift the foot over the obstacle and allow walking to continue. During the stance phase, a stimulus delivered to the foot causes the leg to push down harder, resulting in a shorter stance phase. Although these actions are opposite, both facilitate the continuation of locomotion.

Robotics has adopted the term “reflex” from the biological literature, but in both biology and robotics the precise definition of the term varies from study to study. Most researchers in robotics use the term to mean a quick response triggered by a sensory input. Some require the reflex to be open loop and independent of subsequent sensory input [4], [5]. Others apply the term more loosely to describe actions that are performed with feedback until a terminating sensory input occurs [6]. In some cases reflexes refer to general-purpose actions [7], [8], and in others they refer to actions taken to correct errors or to compensate for disturbances or transitions [9].

Boone and Hodgins [10] proposed strategies to recover a modeled biped from slipping on slippery surfaces with low friction coefficients. They have also addressed the problem of tripping on obstacles with different heights. Their approach employs programmed high-level responses to errors during locomotion in a complex dynamic environment. The hopping biped robot model they used could sustain its original gait and recover from falling on surfaces with friction coefficients as low as 0.025.

Park and Kwon [11] used a reflex control method for biped robots to quickly recover their posture from a slip soon after a detection of a slip and continue to walk with the desired trajectory. Impedance control was used in their control algorithm which enhanced the stability over slippery areas.

Wong and Orin [6] applied reflex control to provide control against slippage and to place the leg of their (ASV) Adaptive Suspension Vehicle in place.

Park and Kwon [11] claim that artificial environments are made suitable for human beings. Therefore, robots which are to live closely with human beings in the same space must have ability to adapt to this man-made environment. Biped robots are excellent choices in this respect. However, they could be easily tipped over or slip on a slippery surface, like human counterparts. As far as slipping of biped robots on a slippery surface is concerned, it is very important that biped robots have a capability to balance themselves and to avoid fall-downs due to slipping, since damages due to a fall-down on a slippery surface could be very costly. In some situations, a human can avoid fall-downs and quickly recover his or her posture using the instinct reflex on a detection of slipping. For the biped robots walking with a normal speed and a stride, there has been no proposed method to overcome foot slipping. Their work proposes a simple but effective method to recover the

robot posture from sudden slipping and to continue walking with the desired trajectory. It is based on the reflex control and the on-line *ZMP* compensation. When a robot enters into a slippery area, it may change its locomotion pattern in many different ways. For example, it may intentionally slide like a skater on ice, or shorten the locomotion step and reduce the locomotion speed. However, they pay no attention to the schemes to change locomotion patterns, but put their main focus on successfully walking on a slippery surface with a normal speed and stride.

It is assumed that the controller does not have information on the slipperiness of the surface a *priori* and thus the biped robot suddenly enters onto a slippery surface with low friction. When the foot of the swinging leg lands and makes a contact with the slippery surface, slipping of the foot could occur. However, when the impedance control is applied such that the landing feet behaviors according to the desired impedance characteristics, the locomotion becomes very robust and slipping phenomena were significantly suppressed without any separate control scheme to avoid slipping. The behavior of the biped robot in this situation is very similar to that of human stepping forward cautiously. It is assumed

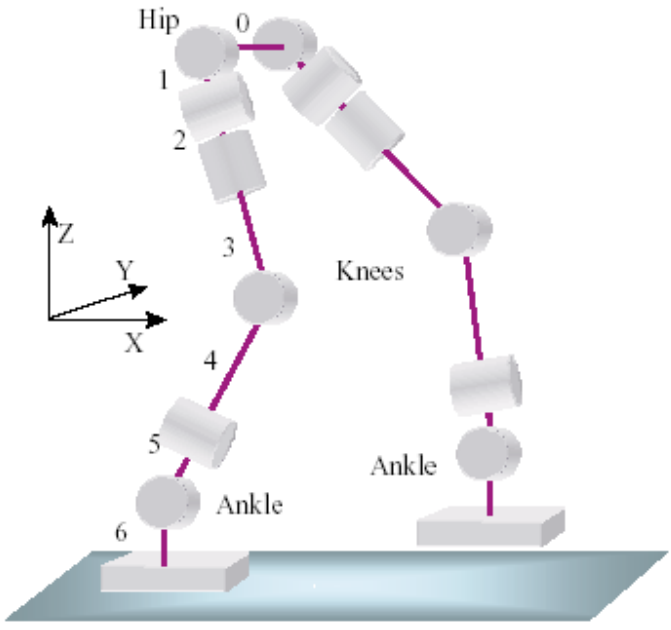

Figure 2.1. A 12-DOF biped robot in [11]

that the walking speed and the locomotion step of the biped robot is not too fast and too large. In fast locomotion with a large size of locomotion step, even the impedance controller cannot avoid slipping and falling-downs.

In the next step, however, the supporting leg is on the slippery surface and a weight shift due to the swing of the free leg could cause slipping. Since a shift of a significant amount of load of the foot on the ground occurs when a single supporting phase starts, the biped robot is very susceptible to slip at the beginning of a single support phase.

Reflex control in [11] consists of three phases: Detection of a slip at the feet, fast adaptation, and the recovery to the normal locomotion pattern. When the horizontal acceleration of the foot on the ground is above a certain level, it is assumed that slipping occurs at the foot. In the fast adaptation phase, reflex control inputs are applied to the biped robot in efforts to prevent further slipping. If further slipping is successfully prevented by the reflex control law, the biped robot returns to the original locomotion pattern in the recovery phase. The schematic diagram of the proposed architecture for the reflex control is shown in Figure 2.2.

The reflex control methods are as follows. First, the hip link of the biped robot is

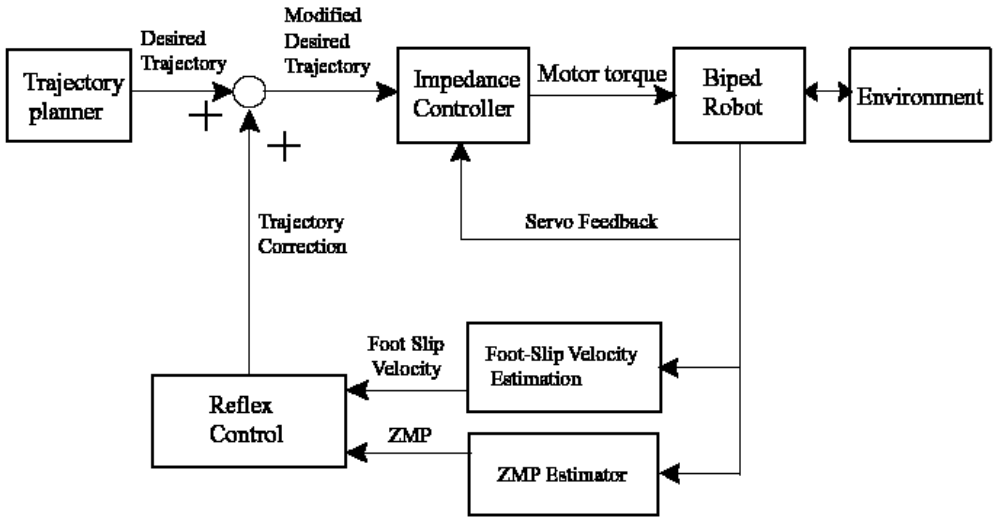

Figure 2.2. Architecture of the proposed control in [11]

lifted up vertically. A positive acceleration of the hip link in the vertical direction would increase the contact force between the ground and the foot on it. Second method is to move the foot of the swinging leg toward the supporting leg in the horizontal direction. The following method to shift the foot of the robot into the horizontal direction keeps the balance of the robot by moving the foot of the swinging leg into the direction for the foot of the supporting leg to slip. When the horizontal acceleration of the foot is small and below a certain level, the controller tries to get back to the original locomotion trajectory by generating a recovery trajectory. The recovery trajectory connects the current configuration of the biped robot with the original trajectory of the biped robot. A 3rd-order interpolating polynomial is used to recover the original trajectory in the vertical direction, and a 2nd-order interpolation polynomials in the lateral direction.

When only reflex control methods described so far are applied to the biped robot, it may be able to overcome slipping quickly in many cases. However, in some other cases, the control could run out of the elevation space of the hip link or the movement space of the swing leg due to the kinematic limits in the robot joints. The problem becomes very realistic and serious especially when the stride of locomotion increases. With a large value of locomotion stride, the range of the elevation of the hip link is further limited.

To cope with these situations, [11] proposes an additional reflex control algorithm based on the online *ZMP* compensation. This method is to shift the hip link in a horizontal direction such that the *ZMP* of the robot moves into the safety region. The hip link is moved by changing its desired acceleration depending on the deviation of the *ZMP* from the boundary of its safety region. A series of computer simulations of locomotion of a 12-DOF biped robot show that the proposed control help the robot keep its balance and recover its posture even when it steps abruptly and unexpectedly on a slippery surface the coefficient of which is as low as 0.3.

Boone and Hodgins [10] studied slipping and tripping reflexes for bipedal robots. Their work explores strategies that would enable legged robots to respond to two common types of surface errors: Slipping and tripping. They propose a set of reflexes that would permit the robot to react without modeling or analyzing the error in detail. They presented simulation trials for single-slip tasks with varying coefficients of friction and single-trip tasks with varying obstacle heights. They claim that most legged robots lack the control

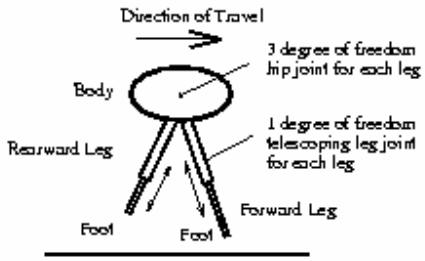

Figure 2.3. Biped structure in [10]. The simulated biped robot consists of a body and two telescoping legs. Each leg has three degrees of freedom and the hip and a forth degree of freedom for the length of the leg.

technique that would allow them walk on relatively simple rough terrain like stairs, curbs, grass and slopes.

The simulated robot used in their research is based on a planar bipedal robot in [12], [13]. The simulated robot is three dimensional and has three controlled degrees of freedom at each hip and one for the length of each leg.

The controller achieves dynamically stable, steady state running by decomposing the control problem into three largely decoupled subtasks: Hopping height, forward velocity and body attitude.

Figure 2.4. explains the slipping reflex strategies they proposed, namely, front leg repositioning, rear leg repositioning, and stable triangle recovery. In front leg repositioning, the front leg is lifted and repositioned in a more vertical impact. In rear leg repositioning, the rear leg is brought to arrest the fall. The newly planted leg slips upon takeoff, but the step is successful because the body attitude is not disturbed significantly. The robot is able to continue running. In the stable triangle recovery, the robot forms a stable triangle after detecting a slip. Although the legs slip just prior to liftoff, the control system is able to recover because the slip is symmetric and occurs at the end of the step.

Figure 2.5. explains the trip strategies they proposed, namely, front lift, rear lift, and rear pull trip responses. In the front lift trip response, the front leg is lifted and repositioned to achieve a better foothold. In the rear lift trip response, the rear leg is lifted and repositioned to achieve a better foothold. Finally in the rear pull trip strategy, when a leg hits an obstacle, while swinging forward, it is pulled back to allow it to clear the obstacle.

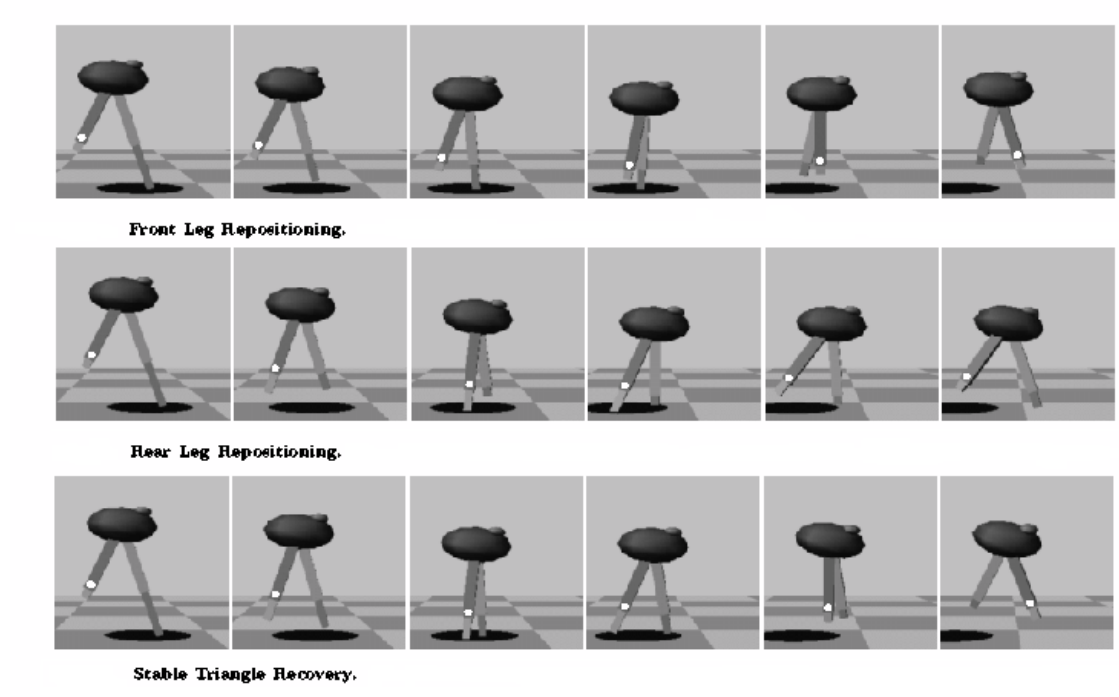

Figure 2.4. Reflex strategies in [10] for slipping.

By using their simulation test bed, the concluded that these slipping and tripping reflexes are robust despite their minimal sensing requirements. Without determining friction or obstacle properties, without modeling the surface, and without online planning, the reflexes enabled the robot to continue running under many circumstances.

There is a number of other interesting studies reported in literature on the biped gate adaptation and reflex strategies [14-24].

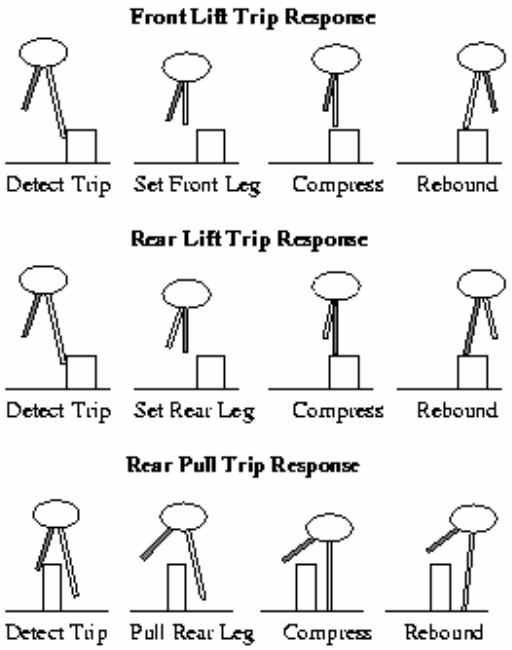

Figure 2.5. Trip recovery strategies in [10]. After a trip has detected, one of the legs is repositioned in an attempt to contact the top surface of the obstacle or avoid it entirely.

CHAPTER 3

PLANT AND THEORY

3.1. The Work Frame

The control strategies studied in this thesis are designed for a twelve degrees of freedom 3D simulated biped robot. The simulation is based on the extension of the results in [25] to tree structured free fall manipulators. The dynamics computations are very similar to the ones in [26] except that a spring and damper based ground contact model is employed in the presented thesis. Each leg has six degrees of freedom: Three at the hip, one at the knee, and two at the ankle. The legs weigh 15 kg each. The body weighs 50 kg. Hence, the weight of the whole robot is 80 kg. Figure 3.1. shows the shape of the robot used in the animation.

The world frame is shown in Figure 3.1, behind the robot. The *x*-axis is in the direction of motion of biped, *y*-axis is the lateral one, and the *z*-axis shows up.

The robot joints are controlled independently by PID controllers. The robot walks by following joint trajectories generated from Cartesian space foot trajectories. Closed form analytic inverse kinematics relations are employed to obtain joint reference positions from Cartesian references. The foot Cartesian trajectories are expressed with respect to a coordinate frame attached to the body coordinate frame.

The orientations of the feet are always kept parallel to the body coordinate frame and hence the reference generation simplifies to finding reference *x*, *y* and *z* trajectories for the two feet.

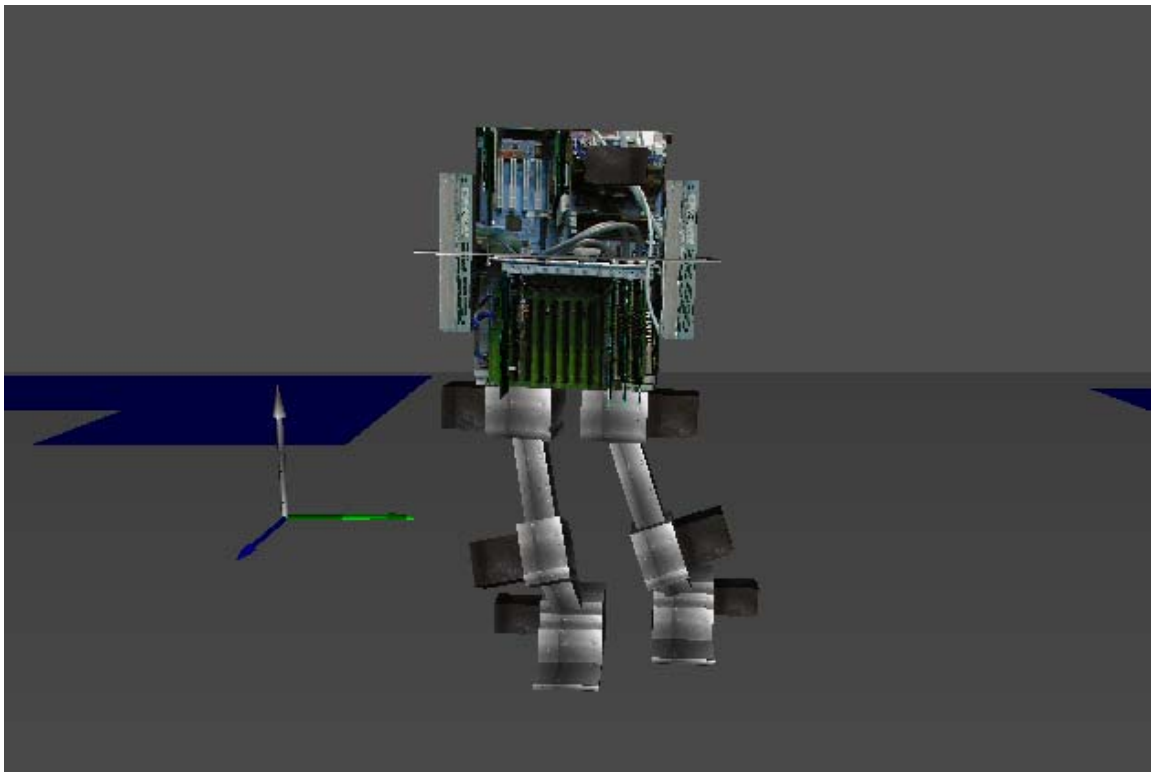

Figure 3.1. The twelve dof bipedal robot used in simulation

3.2. The Walk Pattern Trajectories

The walking pattern of the robot is determined by the trajectory followed by the feet with respect to the body frame. Each foot position is completely determined by x, y, and z coordinates in the body frame. The orientation of the feet is constant in the body frame. The equations for the reference trajectory of the coordinate frames attached to the right and left feet, X_r, Y_r, Z_r, X_l, Y_l, and Z_l are given below.

$$X_r = x_{ref\_as} + \begin{cases} b_s & t < \frac{t_{ds}}{2} \\ b_s \cdot cos\left((t - \frac{t_{ds}}{2}) \cdot \frac{\pi}{t_{ss}} \right) & \frac{t_{ds}}{2} \le t < \frac{t_{ds}}{2} + \mathrm{t_{ss}} \\ -b_s & \frac{t_{ds}}{2} + t_{ss} \le t < \frac{3}{2} t_{ds} + t_{ss} \\ -b_s \cdot cos\left((t - \frac{3}{2} t_{ds} - t_{ss}) \frac{\pi}{t_{ss}} \right) & \frac{3}{2} t_{ds} + t_{ss} \le t < \frac{3}{2} t_{ds} + 2t_{ss} \\ b_s & \frac{3}{2} t_{ds} + 2t_{ss} \le t < 2t_{ds} + 2t_{ss} \end{cases} \quad (3.1)$$

$$X_l = 2 \cdot x_{ref\_as} - X_r \quad (3.2)$$

$$Y_r = y_{ref\_as} - w_{of} + w \cdot sin(t \cdot \frac{\pi}{t_{ds} + t_{ss}}) \quad (3.3)$$

$$Y_l = y_{ref\_as} + w_{of} + w \cdot sin(t \cdot \frac{\pi}{t_{ds} + t_{ss}}) \quad (3.4)$$

$$Z_r = -h_b + \begin{cases} 0 & 0 \le t < \frac{3}{2} t_{ds} + t_{ss} \\ \frac{h_s}{2} \left[1 - cos\left((t - \frac{3}{2} t_{ds} - t_{ss}) \cdot \frac{2\pi}{t_{ss}} \right) \right] & \frac{3}{2} t_{ds} + t_{ss} \le t < \frac{3}{2} t_{ds} + 2t_{ss} \\ 0 & \frac{3}{2} t_{ds} + 2t_{ss} \le t < 2t_{ds} + 2t_{ss} \end{cases} \quad (3.5)$$

$$Z_l = -h_b + \begin{cases} 0 & 0 \le t < \frac{t_{ds}}{2} \\ \frac{h_s}{2}\left[1-\cos\left((t-\frac{3}{2}t_{ds}-t_{ss})\cdot\frac{2\pi}{t_{ss}}\right)\right] & \frac{t_{ds}}{2} \le t < \frac{t_{ds}}{2}+t_{ss} \\ 0 & \frac{t_{ds}}{2}+t_{ss} \le t < 2t_{ds}+2t_{ss} \end{cases} \quad (3.6)$$

The time t is reset every $2t_{ds}+2t_{ss}$. Figure 3.2 depict the trajectory equations above with the following parameters values:

t_{ss}	= 1.4 *s*	w_{of}	= 0.11 *m*	t_{ds}	= 1 *s*
w	= 0.13 *m*	b_s	= 0.08 *m*	y_{ref_as}	= 0 *m*
x_{ref_as}	= -0.09 *m*	h_b	= 0.5 *m*	h_s	= 0.03 *m*

3.3. The Reflex-Adaptation Technique

The reflex is initiated by a sensing element consisting of eight force sensors mounted on the corners of the feet soles. Hence, the upward (*z* component) of the force is assumed to be known to the robot controller. Because of the dynamic characteristics of the robot in the walking mode, it is expected that the sum of the forces on the feet is not the same all the time and changes according to the velocity of the foot when hitting the ground and the time this process is applied in. In other words, it depends greatly on the impact of the foot with the ground. The sum of the simulated ground forces is shown below in Figure 3.3.

By investigating Figure 3.3, it can be noticed that peaks generated by the impact of the foot with the ground occur at high frequencies. Therefore, filtering the data is a requirement that cannot be avoided. Low-pass filtering will help to eliminate these peaks that occur at high frequencies.

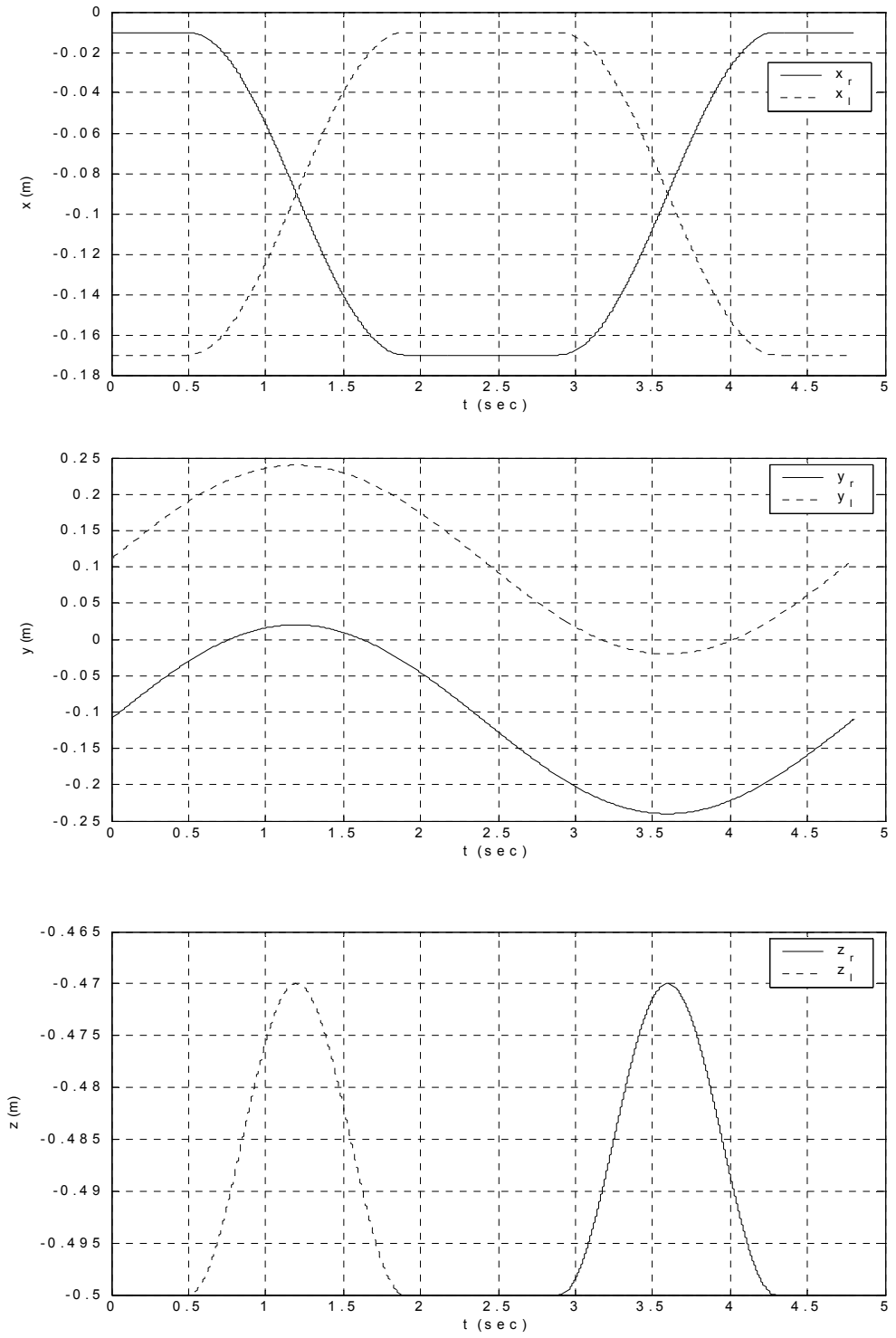

Figure 3.2. *x*, *y*, and *z* trajectory of the walking pattern. This trajectory is for the left and right foot and is with respect to the body frame. The feet are parallel to the body frame and the trajectory is completely determined by the position of the feet frame.

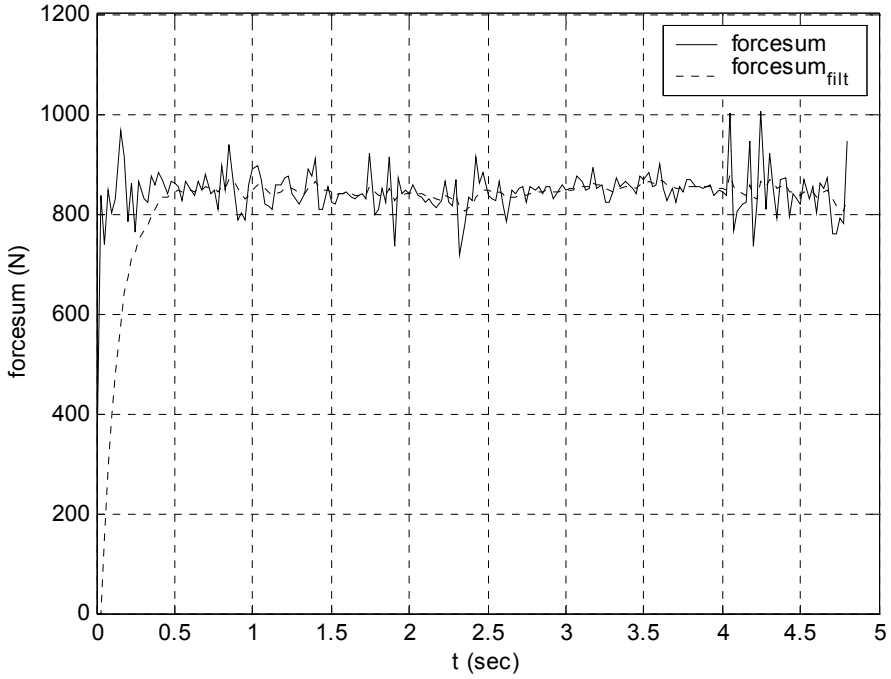

Figure 3.3. Sum of the upward (*z* components) external force exerted by the ground on the robot.

The following low pass filter described by (3.7) is used to filter the upward component of the sum of the impact forces.

$$fs_{filt}^{K+1} = fs_{filt}^{K} \cdot (1-\lambda) + fs^{K+1} \cdot \lambda \qquad (3.7)$$

where *fs* is the *z*-component of the total ground force acting on the feet represented in the body frame; fs_{filt} is its low pass filtered value. λ is a filter parameter, *K* is the simulation cycle at which the values are calculated. It is worth mentioning that (3.7) is applied at the (*K*+1)st simulation cycle. Figure 3.3. shows a typical filtered result with $\lambda = 5 \times 10^{-3}$.

The goal for the biped is to keep on walking after being exposed to a sudden addition of load at the front side of the body. The load is represented by a box at the same height of

the center of mass of the robot, hence there will be no change in the z component of the body. Yet, this addition changes the x-component of the center of gravity.

The center of gravity of the body is set to be 5 cm from the body coordinate frame towards the back of the trunk. If the mass of the load is above a certain value, the biped will fall forward.

One key parameter used in the dynamical model of the robot, consequently, affecting the motion and stability of the robot is the body center of mass vector represented in the body coordinate frame. Adding the load moves the center of gravity of the body forward.

Adding the load changes the inertia matrix and the total weight of the body that are key parameters used in constructing the dynamic equations forming the simulation of the robot. Mathematically speaking addition of the load can be simulated by

$$I_b = \frac{m_b + m_{add}}{m_b} \times I_b \tag{3.8}$$

$$\hat{s} = \begin{bmatrix} \dfrac{20 \times m_{add} - 5 \times m_b}{(m_b + m_{add}) \times 100} \\ 0 \\ \dfrac{H_b}{2} \end{bmatrix} \tag{3.9}$$

$$m_b = m_b + m_{add} \tag{3.10}$$

where m_b is the weight of the body alone, I_b is a 3 x 3 inertia matrix around the body frame

$$I_b = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} = \frac{m_b}{12} \begin{bmatrix} W_b^2 + H_b^2 & 0 & 0 \\ 0 & L_b^2 + H_b^2 & 0 \\ 0 & 0 & L_b^2 + W_b^2 \end{bmatrix}$$

where W_b, H_b, and L_b are the width, height, and length of the body as shown in Figure 3.4. The numerical values of these parameters are as follows: L_b=0.2 m, W_b=0.4 m, H_b=0.5 m, m_b=50 Kg. The body frame is attached to the geometric center of the body as shown in the figure. The center of mass of the body is located 5 cm behind the body frame at half the height and half the width of the body. Hence the initial $\hat{s}$ location is

$$\hat{s} = \begin{bmatrix} -0.05 & 0 & \frac{H_b}{2} \end{bmatrix}^T$$

The load is modeled as a box of dimension 20 cm with a center of mass at the geometric center of the box. The box is mounted at the center of the front side of the body. Hence adding the load will change the $\hat{s}$ to

$$\hat{s} = \begin{bmatrix} \frac{20(cm) \times m_{add}(kg) - 5(cm) \times m_b(kg)}{(m_{add} + m_b)(kg)} \frac{1(m)}{100(cm)} & 0 & \frac{H_b}{2} \end{bmatrix}^T$$

Hence, $\hat{s}$ and m_b are reinitialized according to the mass of the load added.

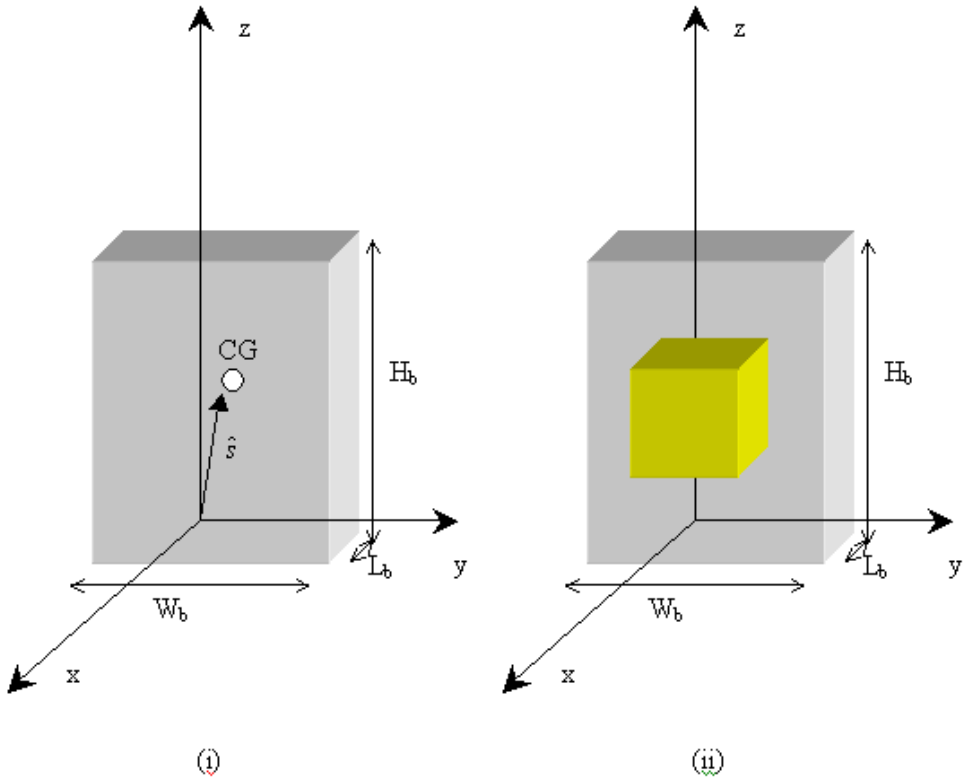

Figure 3.4. (i)Body frame attached to the trunk; center of gravity; and $\hat{s}$ vector (ii)load box of 20 cm length added to the trunk at the front side

Addition of the load is applied all of a sudden at any time. The process does not assume an initial velocity of the load different from the velocity of the body, hence no impact is taken into consideration. Moreover, changing the mass of the load does not imply changing the size or center of mass of the load box. The load can be applied at any time to the robot and in any phase. A load varying from 2.5 kg to 20 kg applied at anytime during a complete left and right step is simulated and resulted in a failure in keeping the stability of the robot.

3.3.1. The Reflex strategy

In the previous section, addition of the load along with sensing this addition was discussed. Here, a strategy of instant recovery is dealt with. This strategy is called reflex because it does not depend on the load added and it does require calculation or specific control action to adapt with. It is a way to bring the robot to a stationary stand after sensing the addition of the load. However, because of the dynamic characteristics of the robot, it is not feasible to reflex instantly during a sampling time because this will cause another source of instability to the robot and one will end up with more complicated problem of instability afterwards. Also, in reality as far as the human beings are concerned, their reflexes take place in a duration of time called the reflex period. Previously the walking pattern was discussed and plotted with respect to the body frame Figure 3.2. Here, the reflex pattern is being considered. Examining and investigating will follow after that.

The trajectory applied during the reflex period is as follows:

$$X_r = x_{r\_s} + \frac{x_{refl\_r}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.11}$$

$$X_l = x_{l\_s} + \frac{x_{refl\_l}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.12}$$

$$Y_r = y_{r\_s} + \frac{y_{refl\_r}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.13}$$

$$Y_l = y_{r\_s} + \frac{y_{refl\_l}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.14}$$

$$Z_r = y_{r\_s} + \frac{z_{refl\_r}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.15}$$

$$Z_l = y_{r\_s} + \frac{z_{refl\_l}}{t_{refl\_per}} \cdot t \qquad 0 \le t < t_{refl\_per} \tag{3.16}$$

Equations (3.11)-(3.16) are simple ramp trajectories bringing the feet position from start state to the end state. The following equations illustrate the end state of the reflex trajectory;

$$\begin{aligned}
x_{refl_r} &= -0.01 \quad \Rightarrow x_{r_e} = x_{r_s} - 0.01 \\
x_{refl_l} &= 0.01 \quad \Rightarrow x_{l_e} = x_{l_s} + 0.01 \\
y_{refl_r} &= -w_{of} - y_{r_s} \quad \Rightarrow y_{r_e} = -w_{of} \\
y_{refl_l} &= w_{of} - y_{l_s} \quad \Rightarrow y_{l_e} = w_{of} \\
z_{refl_r} &= -h_b - z_{r_s} \quad \Rightarrow z_{r_e} = -h_b \\
z_{refl_l} &= -h_b - z_{l_s} \quad \Rightarrow z_{l_e} = -h_b
\end{aligned} \tag{3.17}$$

Figure 3.5. depicts a typical 0.2 s reflex initiated at 1.225 s.

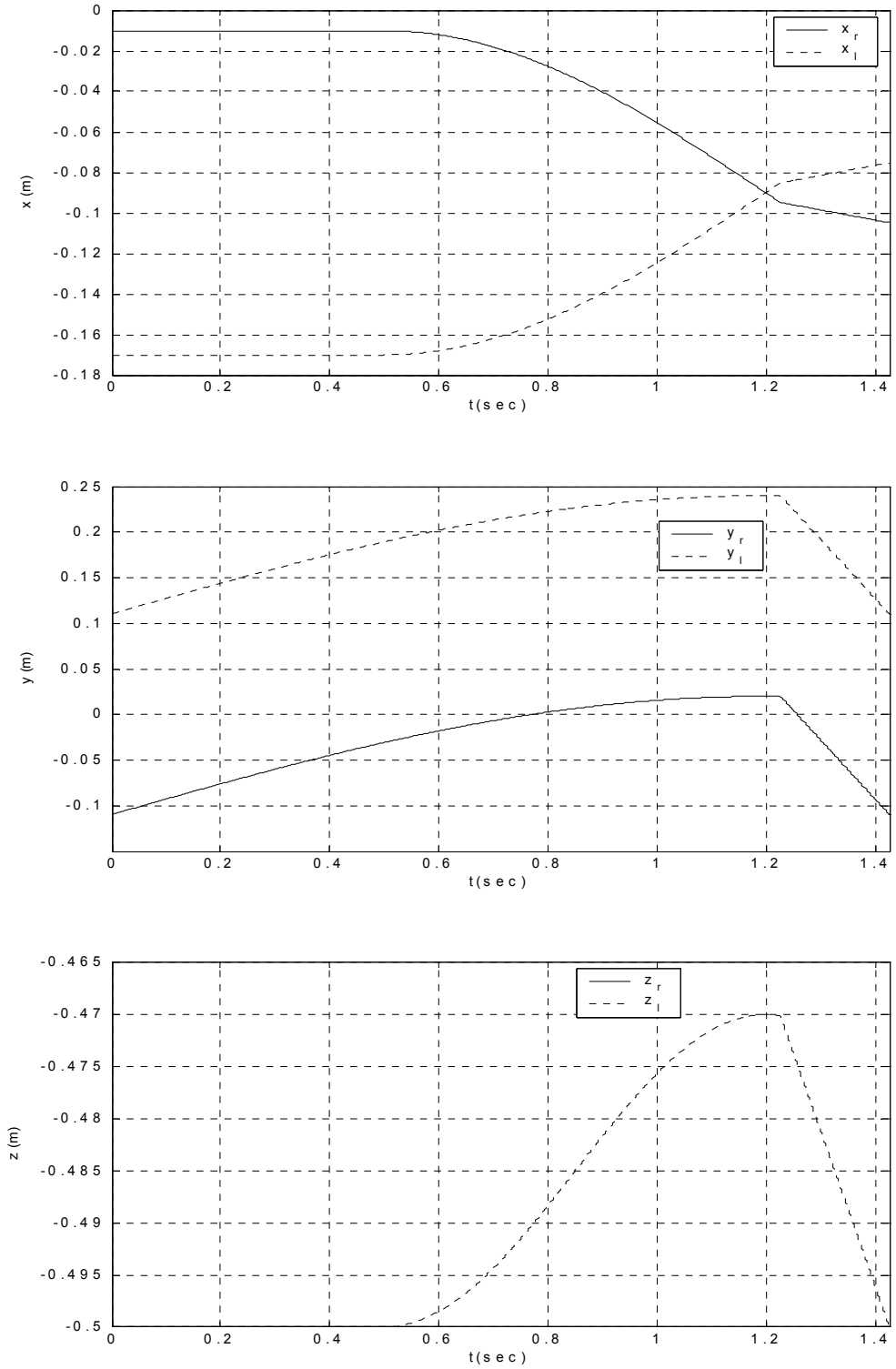

Figure 3.5. Trajectory inset showing the 0.2 s reflex action initiated at 1.225 s

Examining the end states of the trajectories above we see that X_r and X_l are just shifted by 0.01 and –0.01 m respectively. Y_r and Y_l reach their final states; - w_{of} (-0.11 m) and w_{of} (0.11 m) respectively. Z_r and Z_l reach their final value which is – h_b (-0.5 m).

3.3.2. Adaptation and recovery

Reflex action strategy was described in the previous section and as the name implies reflex is a rapid action that is triggered without calculation to lessen the dangerous consequence of an error that has already taken place. Therefore, bringing the robot to the stand position is not enough to gain stability and allow the robot walk again as if no error has taken place. In the following, an adaptation action is proposed to prepare the robot to regain its stability configuration based on some calculation related to the center of gravity of the robot as a whole. Then it brings the original trajectory back by interpolating the last data we have and allows durable and safe walk of the robot.

Figure 3.6. shows filtered data for the zero moment point in the x direction ZMP_x. It is a point on the ground where the tendency of the of the robot to fall around an axis passing through it and parallel to the y axis is zero. In the stationary case where there is no motion, ZMP_x is the x component of the center of gravity. Data shown in Figure 3.6. is for a stationary stand position; hence it represents not only the $ZMPx$ but also the x coordinate of the center of gravity. Here the original data has peaks so it is preferred to filter the data in order to make use of it. Again a low pass filter is been chosen to have those data filtered. The filter used is described in (3.18) below

$$ZMP_{x,filt}{}^{K+1} = ZMP_{x,filt}{}^{K} \cdot (1-\lambda) + ZMP_{x,filt}{}^{K+1} \cdot \lambda \tag{3.18}$$

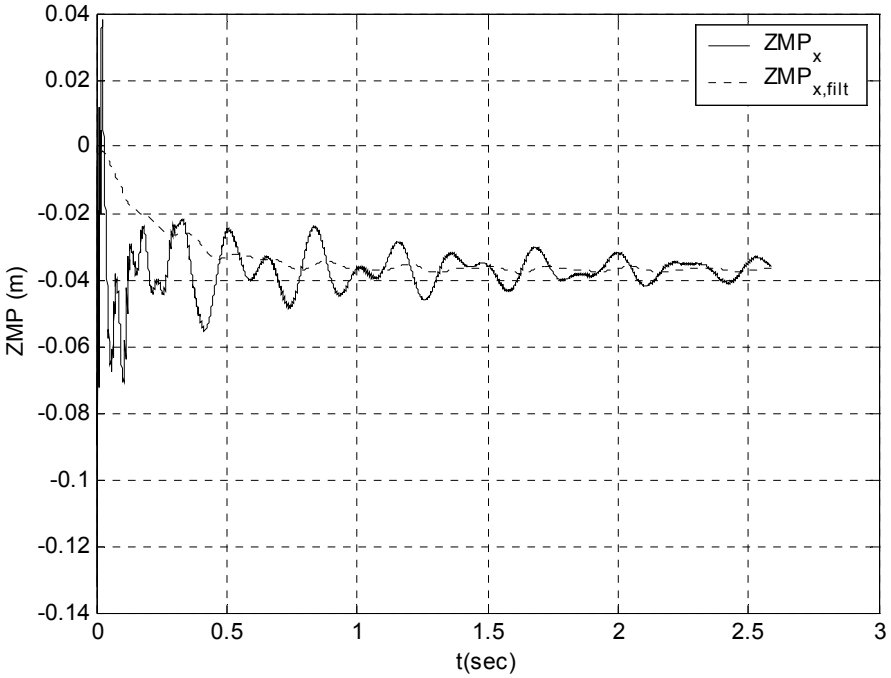

Figure 3.6. ZMP_x represented in the body frame vs. time for a stationary robot

Where ZMP_x is the zero moment point in the x direction; ZMP_{x_filt} is the filtered $ZMPx$ and λ is a filter parameter. K is the simulation cycle at which the values are calculated. Equation (3.18) is applied at the (K+1)st simulation cycle. $\lambda = 5\text{x}10^{-4}$.

Investigating (3.18), the ZMP_x original value associated with the originally balanced (in stand and walk position) robot can be estimated. It is approximately equal to –0.035 m. Adding the load changes the ZMP_x value drastically because this changes the center of gravity position. Hence the problem now is how to regain the ZMP_x original value in order to have the stand and walk balance back. This is applied as follows:

$$x_{ref\_as\_a} = ZMP_{x,filt} - (- 0.035) \tag{3.19}$$

$$x_{ref\_as} = x_{ref\_as} + x_{ref\_as\_a} \tag{3.20}$$

$$x_{r\_e} = x_{r\_e} + \frac{x_{ref\_as\_a}}{\left(\frac{t_{x\_ref\_as}}{\Delta t}\right)} \tag{3.21}$$

(3.19) calculates the value that should be added to x_{ref_as}. It is obvious that it should be equal to the difference between the final value of ZMP_{x_filt} and the ZMP_x value for balanced no load added robot. (3.20) calculates the new value of x_{ref_as} that shall be used later when returning back to the original trajectory. (3.21) is used to create a ramp input which has an initial vale or x_{r_e} and a final value of $x_{r\_e} + x_{ref\_a}$ applied in $t_{x\_ref\_as}$ seconds. (3.21) is applied in the time interval:

$$t \in \left[t\_reflex\_period+t\_settling\_time, t\_reflex\_period+t\_settling\_time+t\_x\_ref\_asymmetry\right]$$

Having ZMP_x shifted to the original balanced value, the robot is now ready to perform walking actions. However, before it can return back to its original trajectory, it must fix its original step size again. In the following it is described how this action is to be applied.

$$dif = \frac{\left|x_{r\_e} - x_{l\_e}\right|}{2} \tag{3.22}$$

$$f = \begin{cases} \dfrac{\left|x_{ref\_as} + b_s - x_{r\_e}\right|}{2 \times b_s} & if(x_{r\_e} \geq x_{l\_e}) \\ \dfrac{\left|x_{ref\_as} + b_s - x_{l\_e}\right|}{2 \times b_s} & otherwise \end{cases} \tag{3.23}$$

$$t_{ss\_a} = \frac{t_{ss} \times cos^{-1}(f)}{\pi} \tag{3.24}$$

$$t_{ds\_a} = \frac{t_{ds}}{t_{ss}} \times t_{ss\_a} \tag{3.25}$$

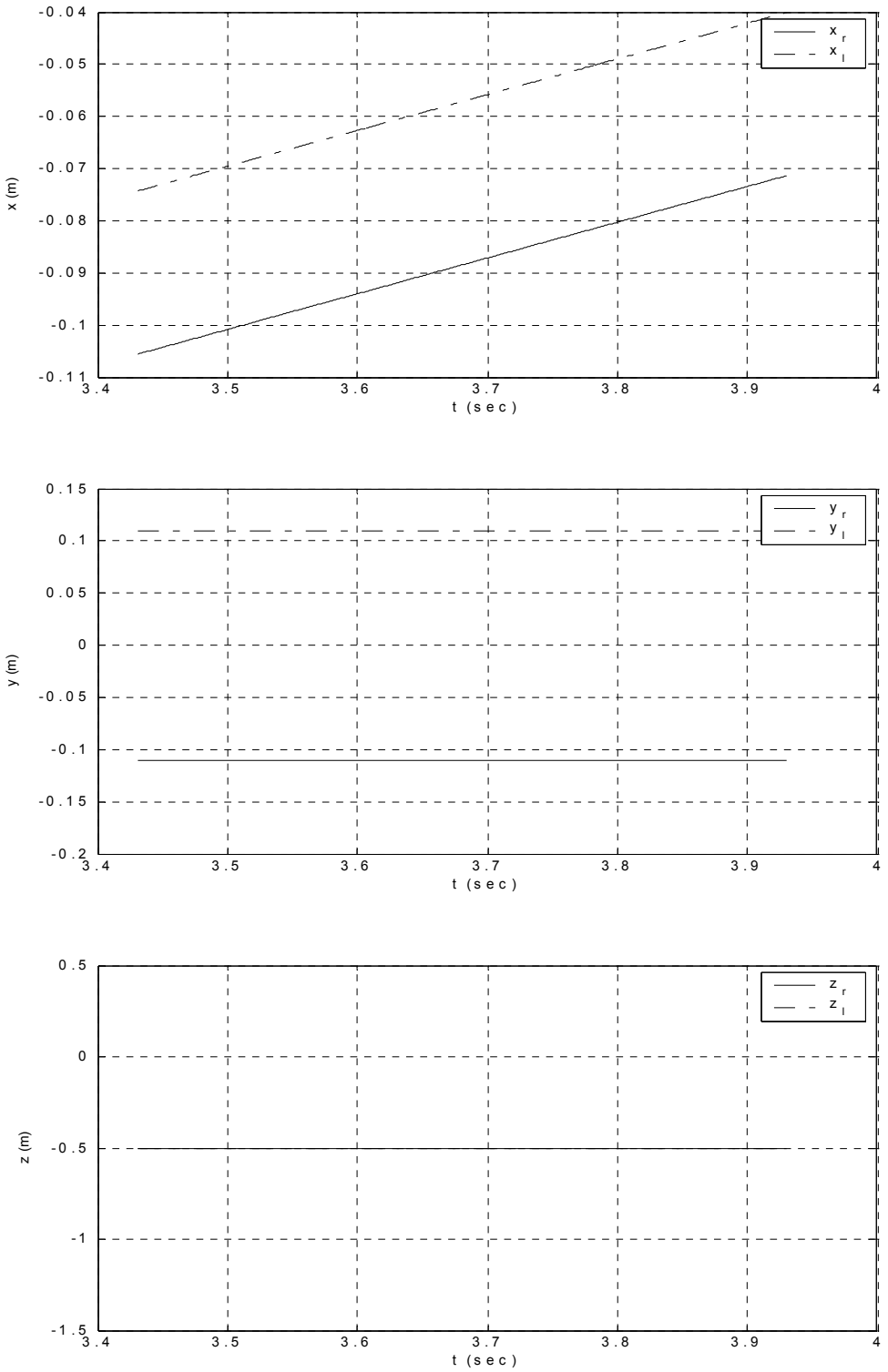

Figure 3.7. Trajectory inset showing the x_{ref_as} started at 3.4305 s and applied in 0.5 s. No change in the y and z coordinates is noticed.

Equations (3.22-3.25) prepare for some values that shall be used later in the adaptation trajectory phase. (3.22) calculates half the distance between the x_{r_e} and x_{l_e} and is called dif. (3.23) gives a factor f the robot shall traverse out of the original walk phase. This value ensures a durable motion during the correction of the step size. Equations (3.24) and (3.25) calculate the time duration in which the corrective step single support and double support phases are applied respectively. To note is that (3.24) makes use of the value of factor obtained in (3.23).

$$X_r = x_{ref\_as} + \begin{cases} x_{r\_e} - x_{ref\_as} & S_1 \\ -dif - \frac{b_s - dif}{2} \times \left[1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + \frac{t_{ds\_a}}{2} + t_{x\_ref\_as} \right) \right) \times \frac{\pi}{t_{ss\_a}} \right) \right] & S_2 \\ dif + \frac{b_s - dif}{2} \times \left[1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + \frac{t_{ds\_a}}{2} + t_{x\_ref\_as} \right) \right) \times \frac{\pi}{t_{ss\_a}} \right) \right] & S_3 \\ -b_s & S_4 \\ b_s & S_5 \end{cases} \quad (3.26)$$

$$X_l = x_{ref\_as} + \begin{cases} x_{l\_e} - x_{ref\_as} & S_1 \\ dif + \frac{b_s - dif}{2} \times \left[1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + \frac{t_{ds\_a}}{2} + t_{x\_ref\_as} \right) \right) \times \frac{\pi}{t_{ss\_a}} \right) \right] & S_2 \\ -dif - \frac{b_s - dif}{2} \times \left[1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + \frac{t_{ds\_a}}{2} + t_{x\_ref\_as} \right) \right) \times \frac{\pi}{t_{ss\_a}} \right) \right] & S_3 \\ b_s & S_4 \\ -b_s & S_5 \end{cases} \quad (3.27)$$

where S_1 to S_5 refer to the conditions under which (3.26)-(3.27) elements are applied. (3.28) explains those conditions:

$$
\begin{aligned}
&S_1 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} \\
&S_2 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} \le x_{l\_e}\right) \\
&S_3 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} > x_{l\_e}\right) \\
&S_4 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} + t_{ss\_a} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + t_{ds\_a} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} \le x_{l\_e}\right) \\
&S_5 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} + t_{ss\_a} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + t_{ds\_a} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} > x_{l\_e}\right)
\end{aligned}
\tag{3.28}
$$

$$
Y_r = y_{ref\_as} - w_{of} + \begin{cases} w \times \frac{f}{2} \times \left(1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + t_{xref\_as}\right)\right) \frac{2\pi}{t_{ds\_a} + t_{ss\_a}}\right)\right) & S_6 \\ -w \times \frac{f}{2} \times \left(1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + t_{xref\_as}\right)\right) \frac{2\pi}{t_{ds\_a} + t_{ss\_a}}\right)\right) & S_7 \end{cases} \tag{3.29}
$$

$$
Y_l = y_{ref\_as} + w_{of} + \begin{cases} w \times \frac{f}{2} \times \left(1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + t_{xref\_as}\right)\right) \frac{2\pi}{t_{ds\_a} + t_{ss\_a}}\right)\right) & S_6 \\ w \times \frac{f}{2} \times \left(1 - cos\left(\left(t - \left(t_{refl\_per} + t_{set\_t} + t_{xref\_as}\right)\right) \frac{2\pi}{t_{ds\_a} + t_{ss\_a}}\right)\right) & S_7 \end{cases} \tag{3.30}
$$

Here, the amplitudes of the trajectory in *y* direction is multiplied by half the value of *f*. This is for the sake of stability. S_6 and S_7 are conditions and are as the following:

$$
\begin{aligned}
&S_6 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + t_{ds\_a} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} \le x_{l\_e}\right) \\
&S_7 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + t_{ds\_a} + t_{ss\_a} \\
&\quad \delta\left(x_{r\_e} > x_{l\_e}\right)
\end{aligned}
\tag{3.31}
$$

$$
Z_r = \begin{cases}
z_{r\_e} & S_1 \\
z_{r\_e} & S_2 \\
-h_b + h_s \dfrac{f}{2}\left(1\text{-}\cos\left(\left(t\text{-}\left(t_{refl\_per} + t_{set\_t} + \dfrac{t_{ds\_a}}{2} + t_{x\_ref\_as}\right)\right)\dfrac{2\pi}{t_{ss\_a}}\right)\right) & S_3 \\
z_{r\_e} & S_8
\end{cases}
\tag{3.32}
$$

$$
Z_l = \begin{cases}
z_{l\_e} & S_1 \\
-h_b + h_s \dfrac{f}{2}\left(1\text{-}\cos\left(\left(t\text{-}\left(t_{refl\_per} + t_{set\_t} + \dfrac{t_{ds\_a}}{2} + t_{x\_ref\_as}\right)\right)\dfrac{2\pi}{t_{ss\_a}}\right)\right) & S_2 \\
z_{l\_e} & S_3 \\
z_{l\_e} & S_8
\end{cases}
\tag{3.33}
$$

The condition S_8 is explained in (3.33) as follows

$$
S_8 : t_{refl\_per} + t_{set\_t} + t_{xref\_as} + \frac{t_{ds\_a}}{2} + t_{ss\_a} < t \le t_{refl\_per} + t_{set\_t} + t_{xref\_as} + t_{ds\_a} + t_{ss\_a}
\tag{3.34}
$$

After the step size regains its own original length, the robot is ready to return to its original walk pattern without any modification on the its trajectory. This can be a very important requirement for possible industrial applications where the robot speed is preferred to be constant throughout the walk. Yet it may be wise to keep the robot stationary before it comes back to walk because of oscillations that the robot may face when heavy loads are applied at more critical instants of time. Hence, it is decided to keep it stationary for half the settling time used in calculating the ZMP_x value. Equations (3.35)-(3.40) describe how this is performed:

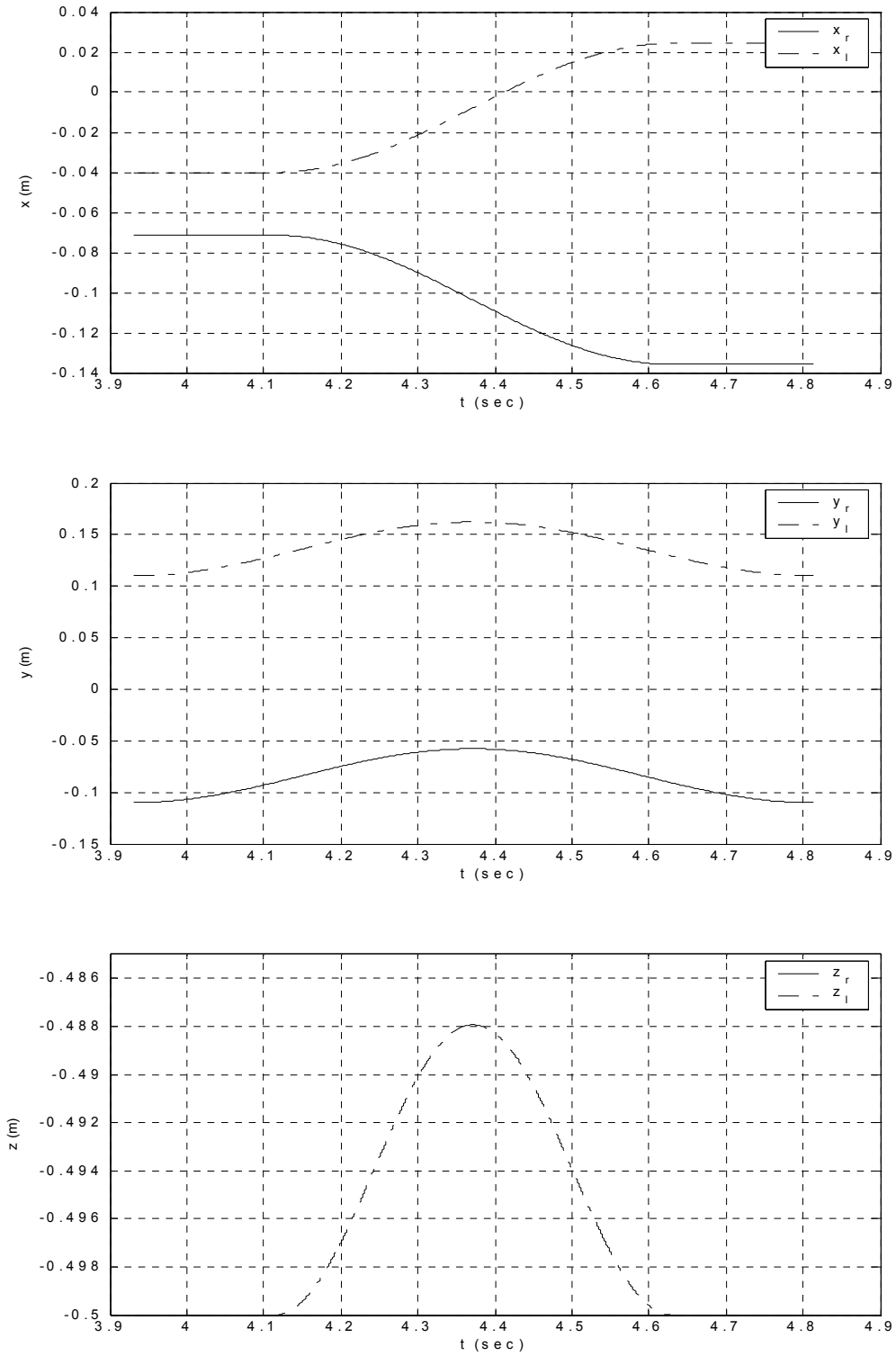

Figure 3.8. Step size adaptation beginning at 3.931 s and ending at 4.814 s

$$X_r = \begin{cases} x_{ref\_as} - b_s & x_{r\_e} \le x_{l\_e} \\ x_{ref\_as} + b_s & x_{r\_e} > x_{l\_e} \end{cases} \tag{3.35}$$

$$X_l = \begin{cases} x_{ref\_as} + b_s & x_{r\_e} \le x_{l\_e} \\ x_{ref\_as} - b_s & x_{r\_e} > x_{l\_e} \end{cases} \tag{3.36}$$

$$Y_r = y_{ref\_as} - w_{of} \tag{3.37}$$

$$Y_l = y_{ref\_as} + w_{of} \tag{3.38}$$

$$Z_r = -h_b \tag{3.39}$$

$$Z_l = -h_b \tag{3.40}$$

Now the robot is ready to follow its original trajectory. Figure 3.9 depicts a complete reflex-adaptation on problem trajectory with a 15 kg load applied at 1.2 s and followed by a complete step.

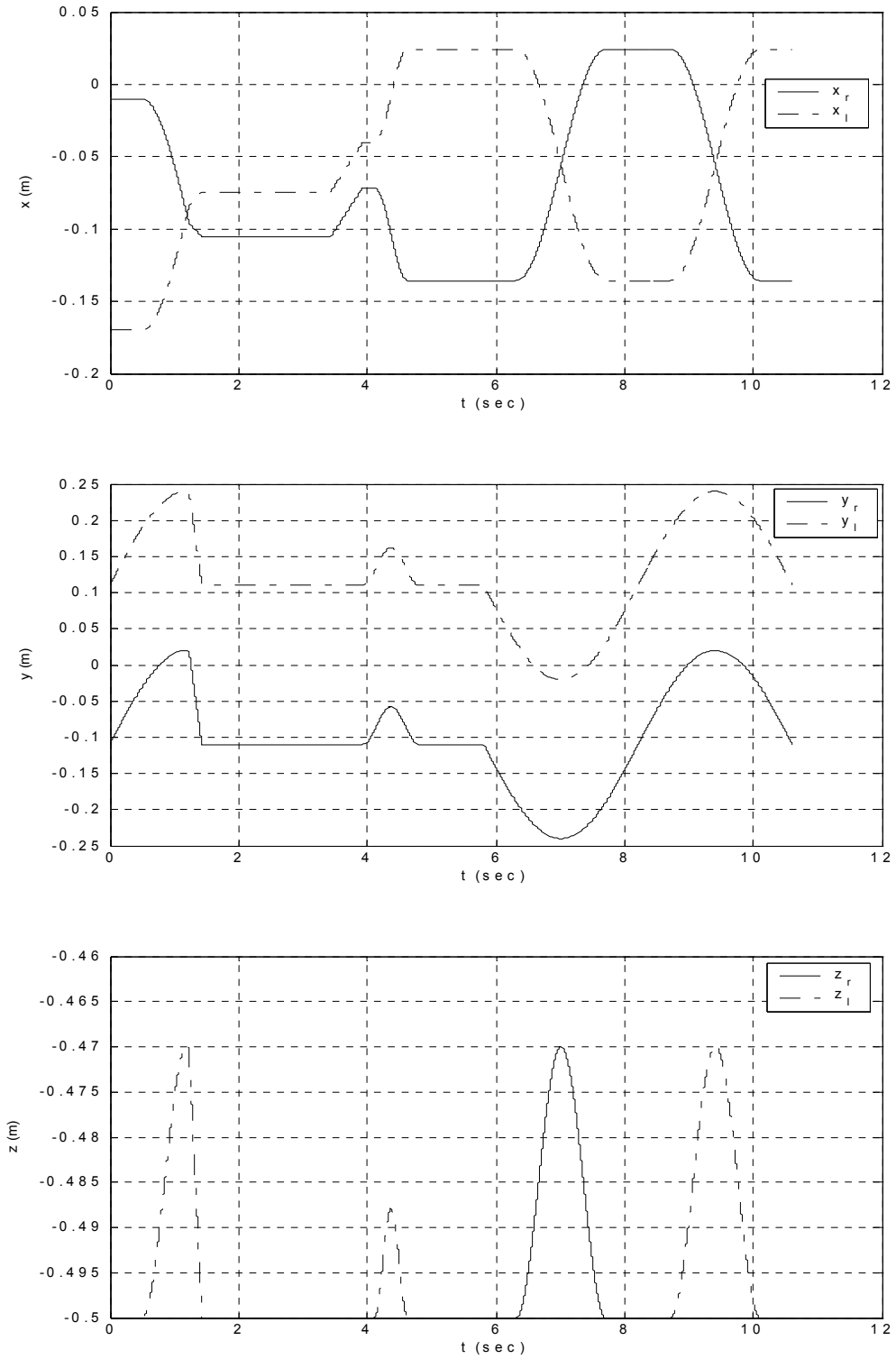

Figure 3.9. A complete reflex-adaptation trajectory and a complete step for a 15 kg load added at 1.2 s.

CHAPTER 4

SIMULATION RESULTS AND DISCUSSION

The main methodology of the reflex and recovery system for the biped robot is discussed in the previous chapter. In this chapter, various simulation results are presented.

A 12 dof model is used to simulate the biped walking robot considered in this thesis, as explained in Chapter 2. The reflex-adaptation technique is tested on the robot in the walk phase. Loads ranging from 2.5 kg to 15 kg with an increment of 2.5 kg were used in the experiments. The experiments were conducted also at different instants of time during the single support and double support periods. The reflex consists of a very short time-duration transition from the single support phase (swing phase) to the double support phase (stance phase) with a direct upright stance. The reflex is applied in 0.1 s. Adaptation of the trajectory and reconfiguration of some of the parameters (ZMP_x) follow after that. Retaining the original step size and trajectory is obtained by interpolating the data with a smooth parameterized trajectory. Refer to Chapter 2 for more details.

4.1. Simulation Examples

In this section, it is demonstrated how the experiments are conducted with detailed results.

In this experiment 5 kg load is added at 1.2 s instant of time. Reflex-adaptation “on” and “off” results are shown in the following.

4.1.1. Reflex-adaptation “off” case

The trajectories to be followed by the robot are shown in Figure 4.1. Figure 4.2. depicts the stability of the robot in terms of the rotation angle around the y axis. This angle is called β and is measured in degrees.

The trajectory shown in Figure 4.1. is a typical one traced by the biped controller. Here, the robot falls down. This is shown clearly in Figure 4.2. where the angle β grows drastically indicating a fall down.

4.1.2. Reflex-adaptation “on” case

This section experiments the reflex-adaptation technique functionality for the problem considered (5 kg load applied at 1.2 s). Figure 4.3 depicts the trajectory input to the controller and to be followed by the robot. Although the load is added at 1.20 s, it takes for the filtered output of the force sensor some time to reach the threshold and trigger the reflex. In this experiment the reflex is triggered at 1.2625 s. The reflex action lasts for 0.1 s and ends at 1.3625 s. A settling time needed to have an accurate reading of the current ZMP_x value lasts for 2 s. During this time the robot performs no motion and follows a stationary trajectory. The settling time ends at 3.3625 s. At the end of the settling stage a read of the filtered ZMP_x value is recorded. A shift of the x_{ref_as} is followed in 0.5 s to reach the ZMP_x value needed for balanced and viable walking. The shift ends at 3.8625 s. Thereafter, interpolating the final data to reach the original desired step size is followed. The time duration in which this process is applied is a function of the ratio of the final obtained step size to the original one. In this case, this duration is computed as 0.9230 s and the recovery phase ends at 4.7855 s. A 1 s period is given to the robot to regain stability and to prepare for the original trajectory that it used to walk with at the beginning. The original trajectory is regained and now ready to be followed. The original walk phase starts at 5.7855 s. This can be seen clearly in Figure 4.3.

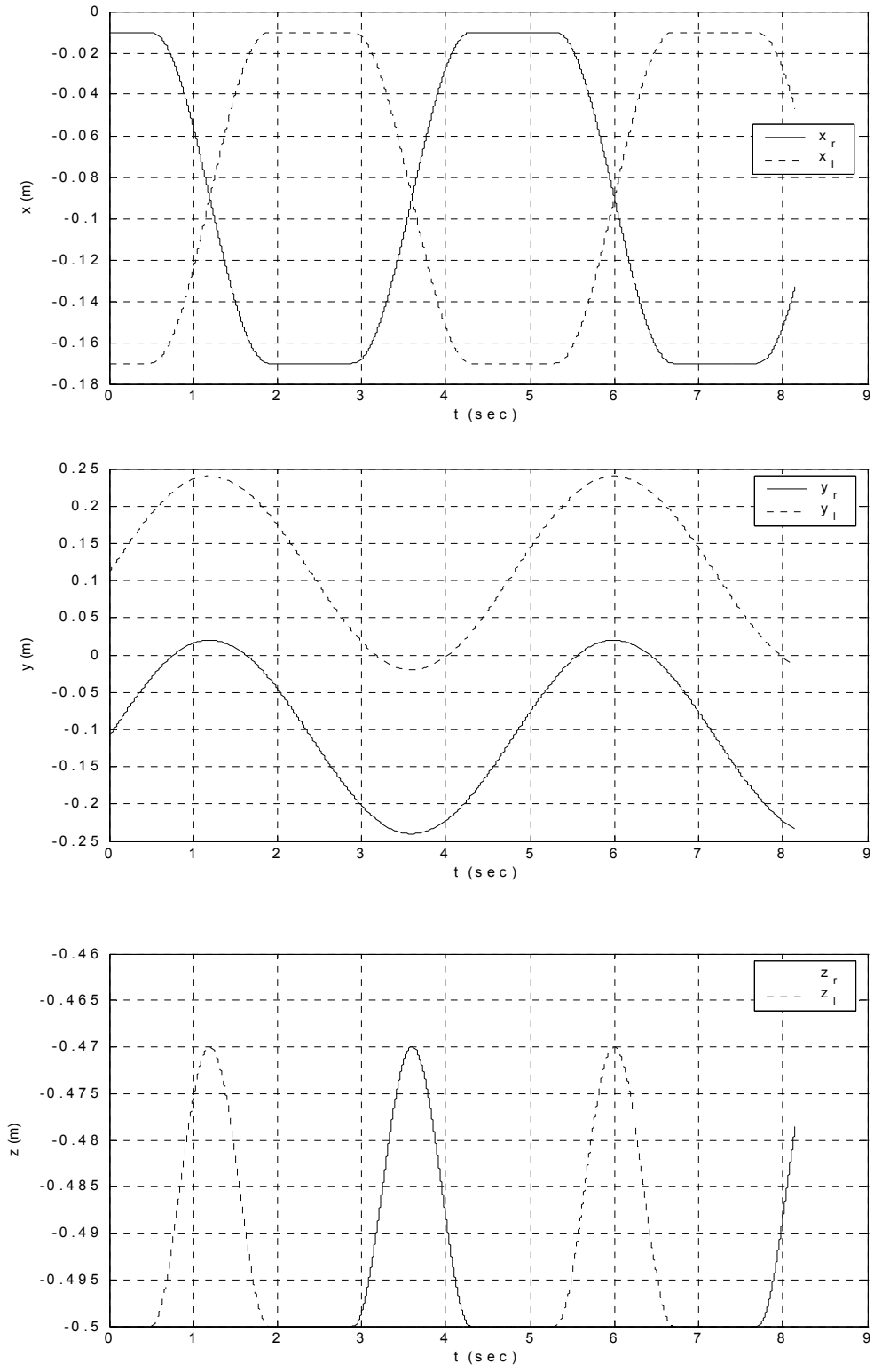

Figure 4.1. Reference generation for reflex-adaptation off, 15 kg at 1.2 s problem

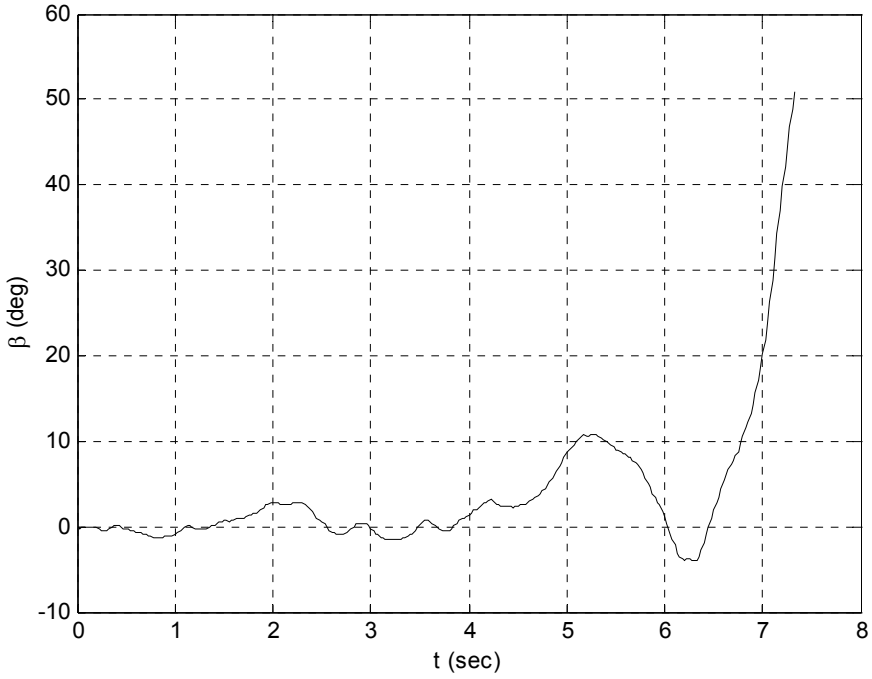

Figure 4.2. Rotation angle around the *y*-axis β for reflex-adaptation off, 15 kg at 1.2 s problem

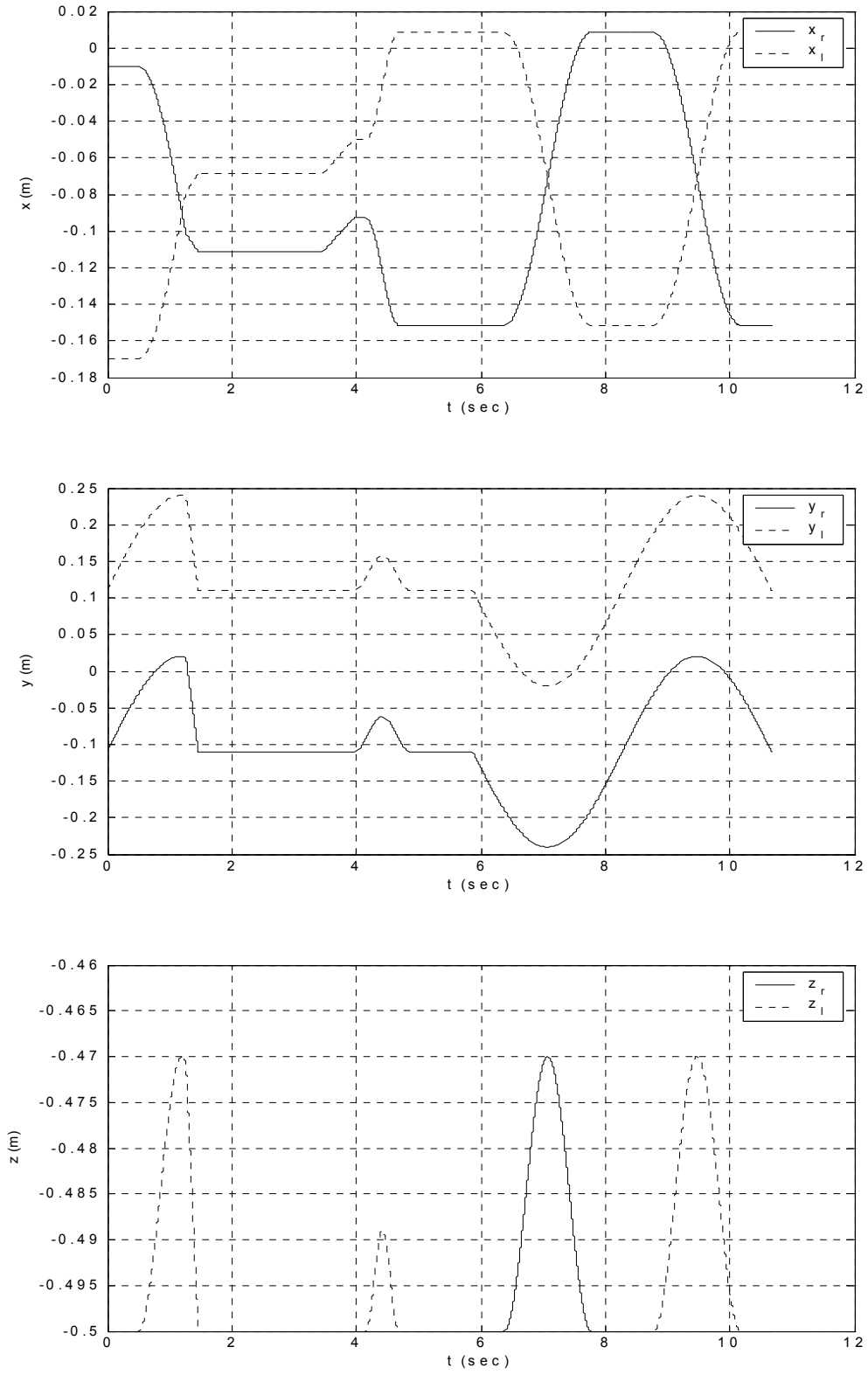

Figure 4.3. Reference generation for reflex-adaptation on, 15 kg at 1.2 s problem

Figure 4.3. shows the balance measure parameter used in this study which is the β angle (angle of rotation around the y-axis). As can be seen from Figure 4.3, this angle does not exceed 2.5 degrees in magnitude which shows the stability and the success of this technique for the problem considered.

This experiment was called successful because the technique helped preventing the robot from falling down. The robot could make two complete steps after the reflex-adaptation method is applied.

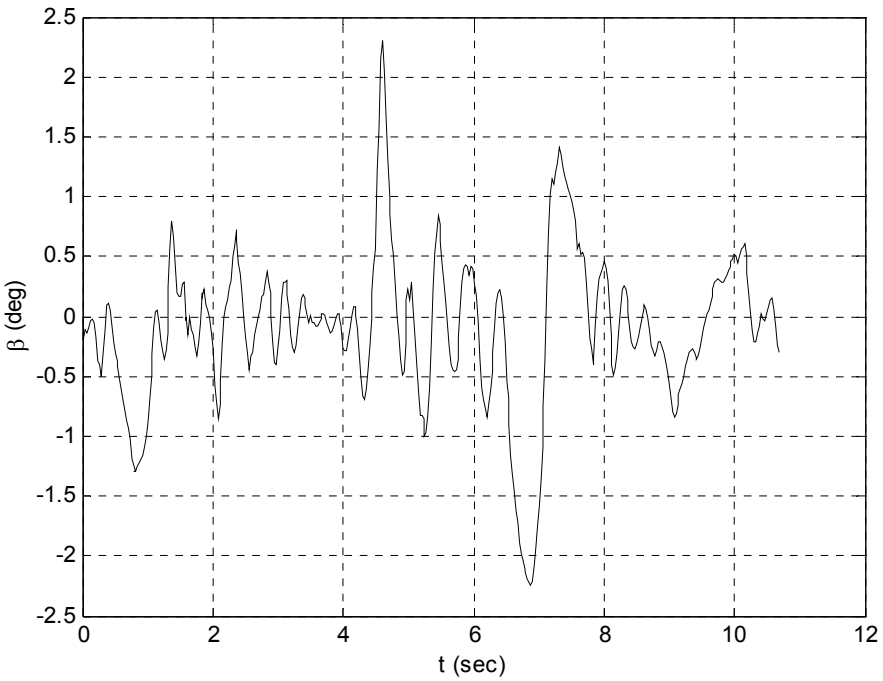

Figure 4.4. Rotation angle around the y-axis β for reflex-adaptation on, 15 kg at 1.2 s problem

4.2. Results with Different Load and Loading Times

Simulation results are carried out for a range of added loads and loading times in the walk cycle. The results of the simulations are as shown in Tables 4.1. and 4.2. Table 4.1. applies the problem with no reflex-adaptation technique on. Table 4.2. shows the results of the reflex-adaptation technique on. A “1” indicates a successful result and “0” indicates failure. A result was called successful if the robot could recover from falling at the end of the step at which the load is added and could continue walking for two complete steps. The result was considered failure otherwise.

Table 4.1. Experiments conducted with reflex-adaptation off. 1 indicates successful experiment and 0 indicates failure.

Load (kg) / Time (s)	2.5	5.0	7.5	10	12.5	15
0.1	1	0	0	0	0	0
0.2	1	0	0	0	0	0
0.3	1	0	0	0	0	0
0.4	1	0	0	0	0	0
0.5	1	0	0	0	0	0
0.6	1	0	0	0	0	0
0.7	1	0	0	0	0	0
0.8	1	0	0	0	0	0
0.9	1	0	0	0	0	0
1.0	1	0	0	0	0	0
1.1	1	0	0	0	0	0
1.2	1	0	0	0	0	0
1.3	1	0	0	0	0	0
1.4	1	0	0	0	0	0
1.5	1	0	0	0	0	0
1.6	1	0	0	0	0	0
1.7	1	0	0	0	0	0
1.8	1	0	0	0	0	0
1.9	1	0	0	0	0	0
2.0	1	0	0	0	0	0
2.1	1	0	0	0	0	0
2.2	1	0	0	0	0	0
2.3	1	0	0	0	0	0
2.4	1	0	0	0	0	0
2.5	1	0	0	0	0	0
2.6	1	0	0	0	0	0
2.7	1	0	0	0	0	0
2.8	1	0	0	0	0	0
2.9	1	0	0	0	0	0
3.0	1	0	0	0	0	0
3.1	1	0	0	0	0	0
3.2	1	0	0	0	0	0
3.3	1	0	0	0	0	0
3.4	1	0	0	0	0	0
3.5	1	0	0	0	0	0
3.6	1	0	0	0	0	0
3.7	1	0	0	0	0	0
3.8	1	0	0	0	0	0
3.9	1	0	0	0	0	0
4.0	1	0	0	0	0	0
4.1	1	0	0	0	0	0
4.2	1	0	0	0	0	0
4.3	1	0	0	0	0	0
4.4	1	0	0	0	0	0
4.5	1	0	0	0	0	0
4.6	1	0	0	0	0	0
4.7	1	0	0	0	0	0
4.8	1	0	0	0	0	0

Table 4.2. Experiments conducted with reflex-adaptation on. 1 indicates successful experiment and 0 indicates failure.

Load (kg) / Time (s)	2.5	5.0	7.5	10	12.5	15
0.1	1	1	1	1	1	1
0.2	1	1	1	1	1	1
0.3	1	1	1	1	1	1
0.4	1	1	1	1	1	1
0.5	1	1	1	1	1	1
0.6	1	1	1	1	1	1
0.7	1	1	1	1	1	1
0.8	1	1	1	1	1	1
0.9	1	1	1	1	1	1
1.0	1	1	1	1	1	1
1.1	1	1	1	1	1	1
1.2	1	1	1	1	1	1
1.3	1	1	1	1	1	0
1.4	1	1	1	1	0	0
1.5	1	1	1	1	0	0
1.6	1	1	1	1	1	0
1.7	1	1	1	1	1	0
1.8	1	1	1	1	1	1
1.9	1	1	1	1	1	1
2.0	1	1	1	1	1	1
2.1	1	1	1	1	1	1
2.2	1	1	1	1	1	1
2.3	1	1	1	1	1	1
2.4	1	1	1	1	1	1
2.5	1	1	1	1	1	1
2.6	1	1	1	1	1	1
2.7	1	1	1	1	1	1
2.8	1	1	1	1	1	1
2.9	1	1	1	1	1	1
3.0	1	1	1	1	1	1
3.2	1	1	1	1	1	1
3.2	1	1	1	1	1	1
3.3	1	1	1	1	1	1
3.4	1	1	1	1	1	1
3.5	1	1	1	1	1	1
3.6	1	1	1	1	1	1
3.7	1	1	1	1	1	0
3.8	1	1	1	1	0	0
3.9	1	1	1	1	0	0
4.0	1	1	1	1	1	0
4.2	1	1	1	1	1	0
4.3	1	1	1	1	1	1
4.4	1	1	1	1	1	1
4.5	1	1	1	1	1	1
4.6	1	1	1	1	1	1
4.7	1	1	1	1	1	1
4.8	1	1	1	1	1	1

4.3.Discussion and Parameter Tuning

This section deals with some of the failed experiments and tries to find solutions for them by either tuning some of the parameters or changing them absolutely.

4.3.1. 15 kg at 1.3 s problem

It is shown in Table 4.2. that the technique considered for 15 kg applied at 1.3 s failed. This is actually because the reflex failed to rescue the robot from falling down. The robot crashed immediately right after the reflex was activated at 1.3325 s. Examining the action of the reflex, it is discovered that if the robot changes its reflex such that x_{refl_r} becomes a positive quantity and x_{refl_l} becomes a negative quantity then the reflex would be able to rescue the robot. Hence, the following parameters are updated as appear in (4.1)

$$x_{refl\_r}=0.01 \quad \Rightarrow x_{r\_e}=x_{r\_s}+0.01 \tag{4.1}$$

$$x_{refl\_l}=-0.01 \quad \Rightarrow x_{l\_e}=x_{l\_s}-0.01 \tag{4.2}$$

This phenomenon can be interpreted as follows. Adding the load to the front side of the trunk increases the moment around the body frame *y*-axis trying to topple the robot forward. If the reflex brings the foot on the ground closer to the body, a decrease in the moment described above is noticed and the likeliness of toppling the robot forward decreases as well.

Figure 4.5. shows the failure and success of the experiments corresponding to the old and new parameters respectively. Figure 4.6. and 4.7. show the *x* trajectories corresponding to the old and new parameters again.

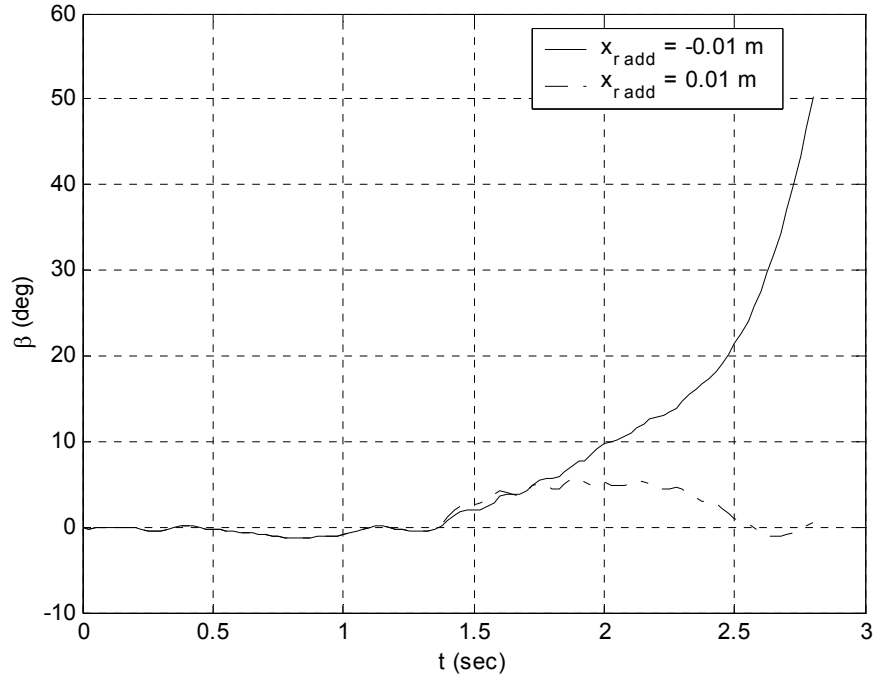

Figure 4.5. β angle for both old and new x_{r_add} for the problem considered.

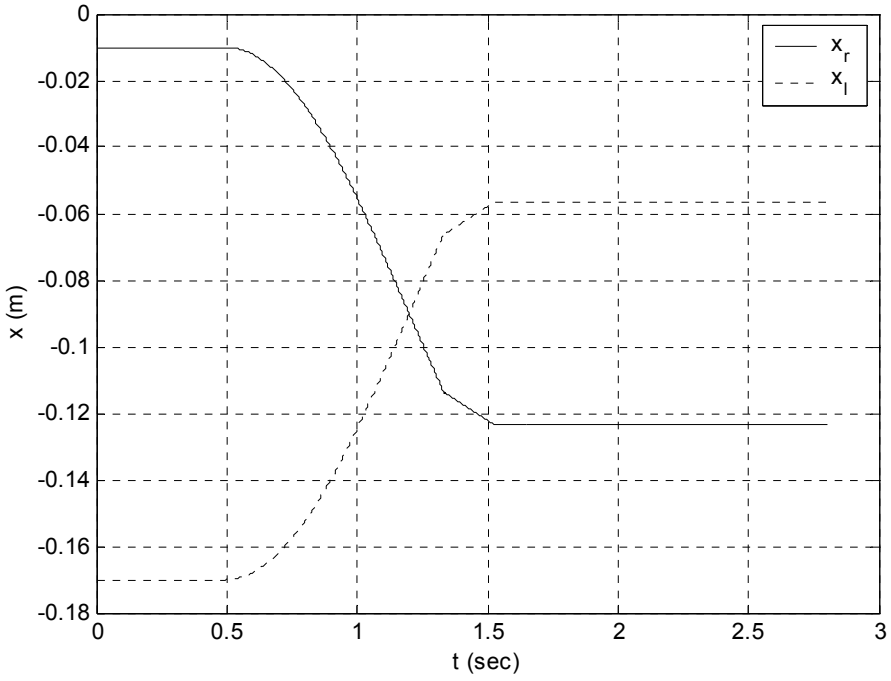

Figure 4.6. Old parameter reflex. Reflex starts at 1.3325 s and ends at 1.4325 s. The right leg which is on the ground is getting far from the body pushing the body forward.

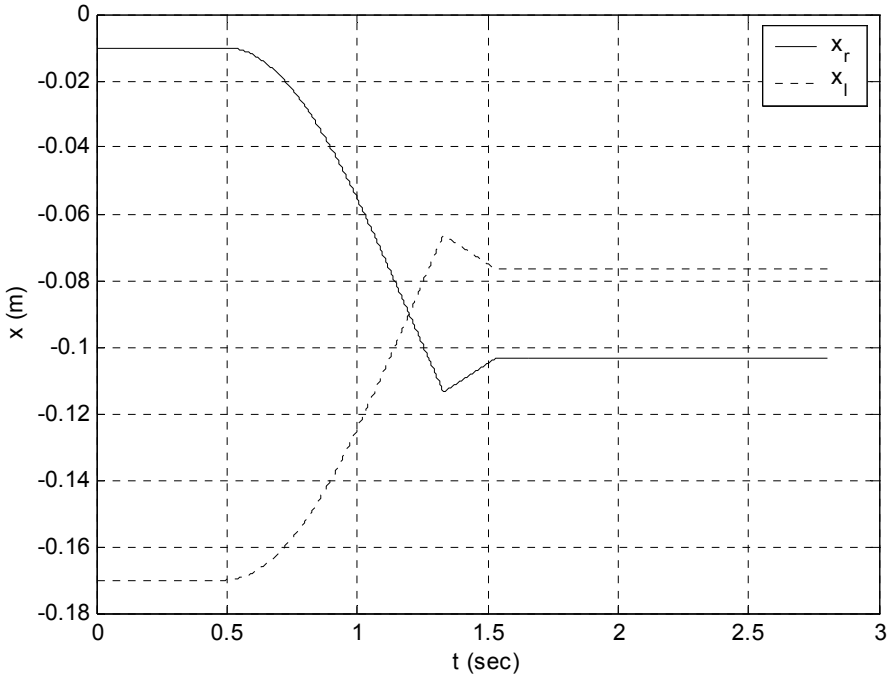

Figure 4.7. New parameters reflex. Reflex starts at 1.3325 s and ends at 1.4325 s. The right leg which is on the ground is getting closer to the body pulling the body backward.

The experiment above proposes the foot on the ground approach the body. This has a pull back effect on the body and decreases the applied load effect.

Although the reflex can now rescue the robot from falling down immediately, the results show that the biped can not go for two normal steps after the adaptation stage finishes. This problem can arise from a wrong shift of the body center. Hence, one may expect a wrong reading of the ZMP_x value. This can be caused by the lack of time to have the biped settle down and come to the steady state. Doubling the settling time period enhances stability so much as can be seen clearly in the following results.

Figures 4.8. and 4.9. graph the β angle and the path trajectory with a 2 s settling time. Figures 4.10. and 4.11. graph the same curves for a 4 s settling time.

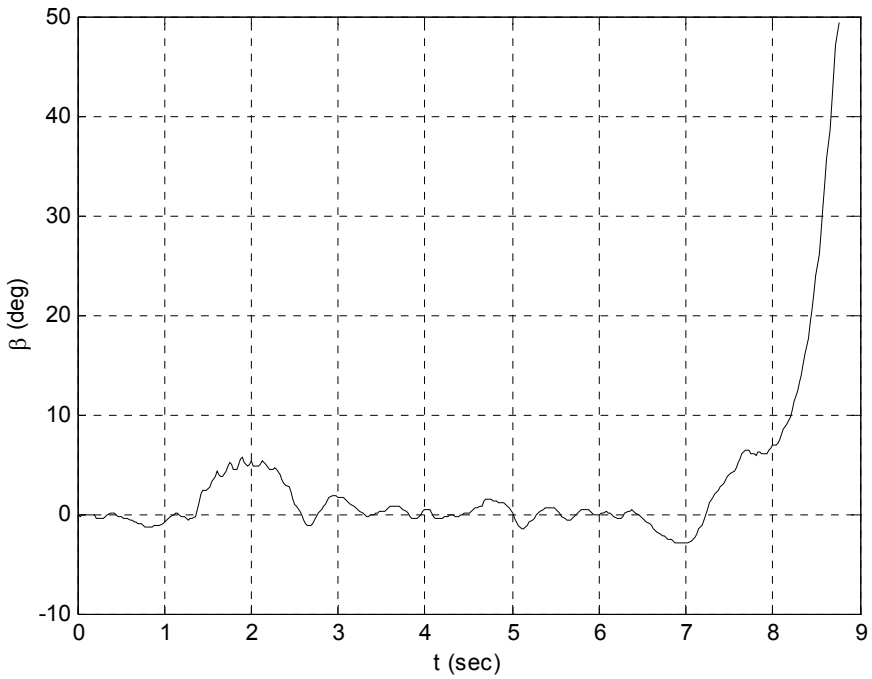

Figure 4.8. β angle for 2 s settling time. The figure shows failure.

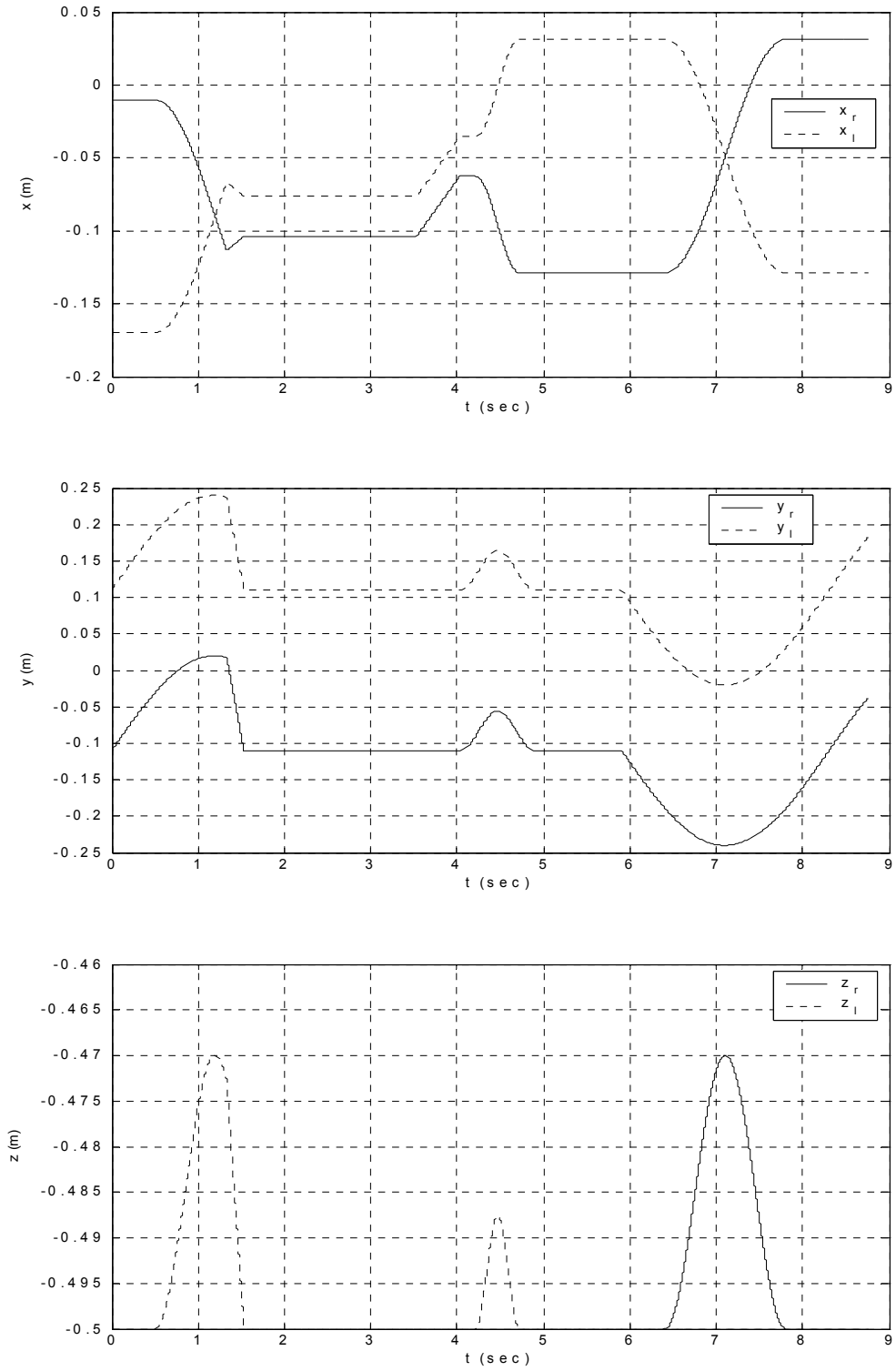

Figure 4.9. Trajectory with settling time is equal to 2 s

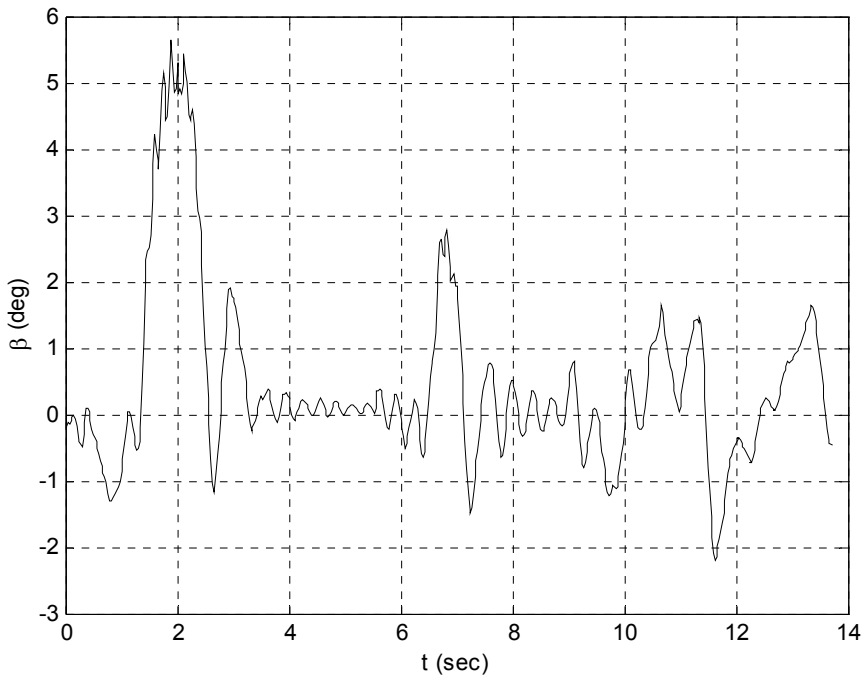

Figure 4.10. β angle for 4 s settling time.

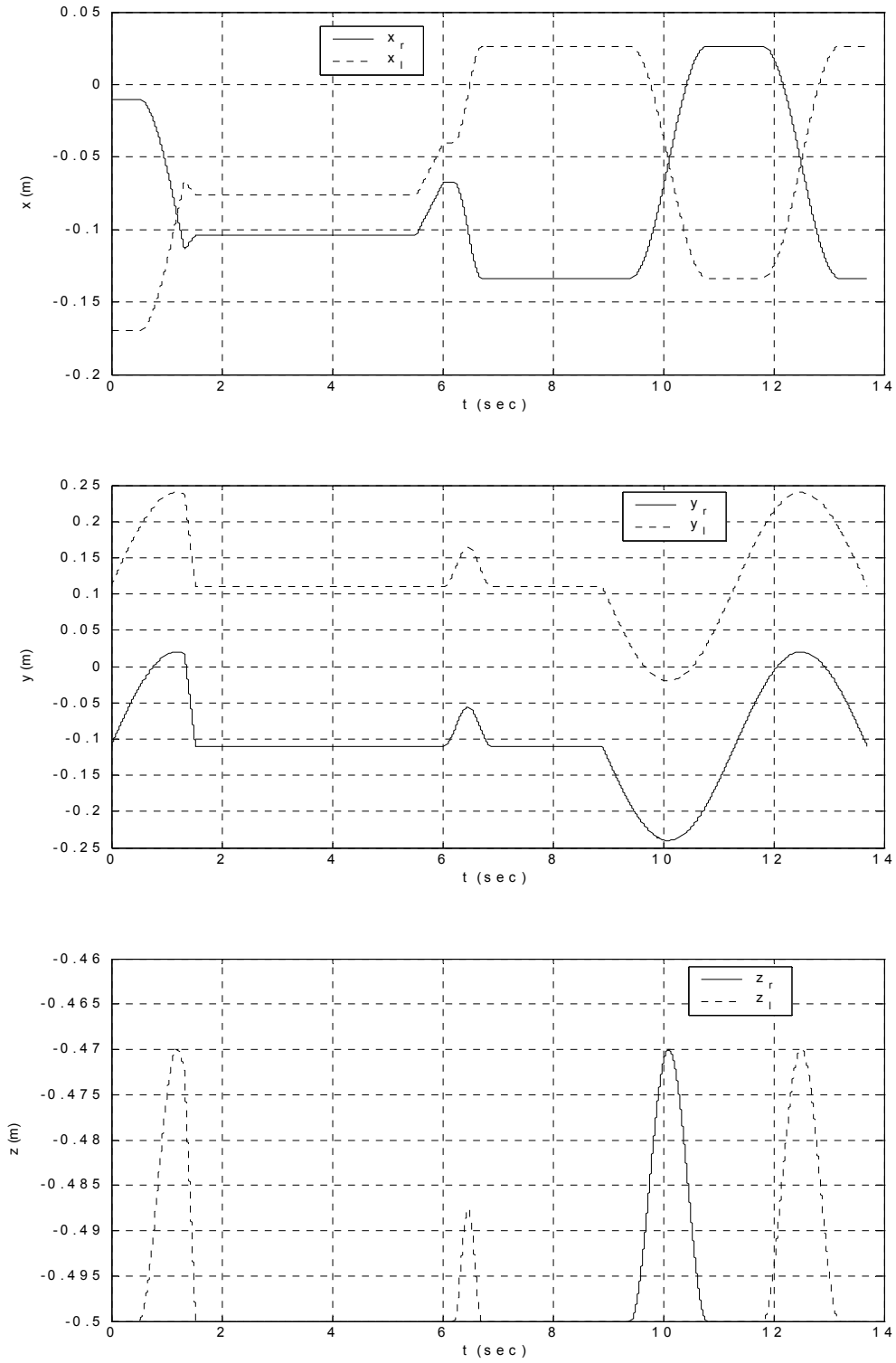

Figure 4.11. Trajectory with 4 s settling time

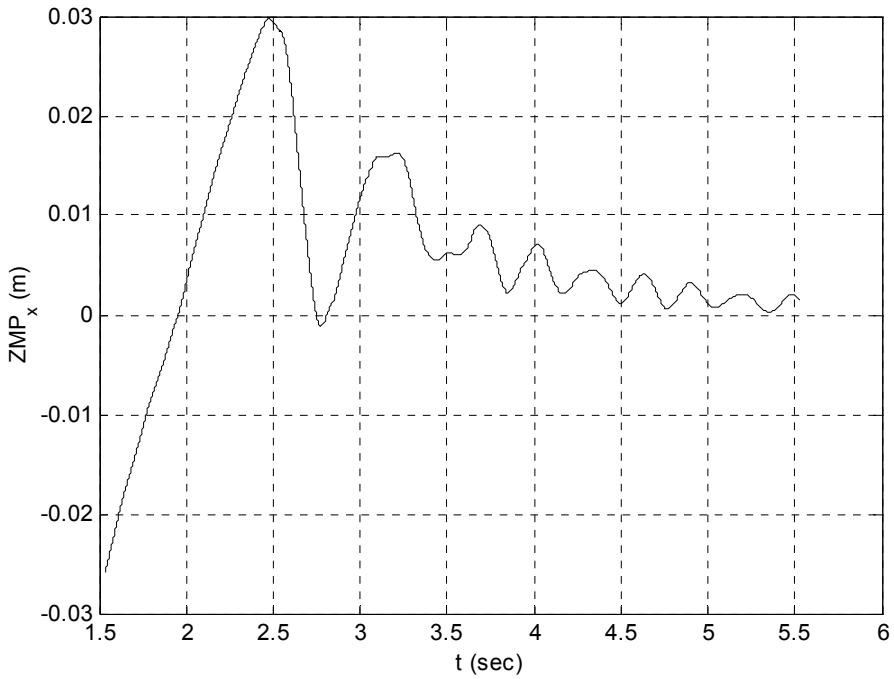

Figure 4.12. ZMP_x vs. time during the settling time.

Figure 4.12. plots the ZMP_x value vs, time from the beginning of the settling time period (1.5325 s). The plot shows the ZMP_x value after 2 (0.0061798 m) and 4 (0.0014766 m) seconds. This might seem a small difference around 0.5 cm but that may affect stability significantly as shown in Figure 4.8. and 4.10. above.

4.3.2. Load applied during the double support phase

Table 4.1. shows that all load addition problems at the double-support phase were successful. A common drawback was discovered in the trajectory proposed in the double-support phase. A 15 kg added at 1.9 s is investigated in Figure 4.13.

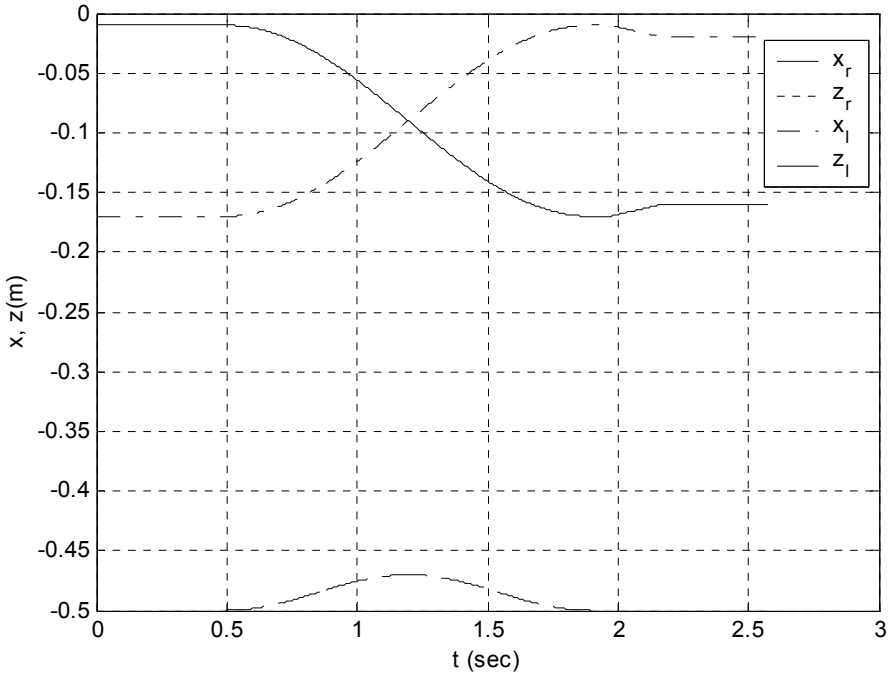

Figure 4.13. *x* and *z* trajectory for a reflex applied at 1.9725 s for a load added at 1.9 s

Examining this trajectory closely at the time the reflex is applied i.e., 1.9725 s to 2.0725 s, it can be noticed that *x* trajectory is changing while both of components of the *z* trajectory are equal to $-h_b$. Equations (4.3) and (4.4) propose a solution for this problem.

$$x_{refl\_r}=0.0 \qquad if\ Z_r=-h_b\ and\ Z_l=-h_b \tag{4.3}$$

$$x_{refl\_l}=0.0 \qquad if Z_r=-h_b\ and\ Z_l=-h_b \tag{4.4}$$

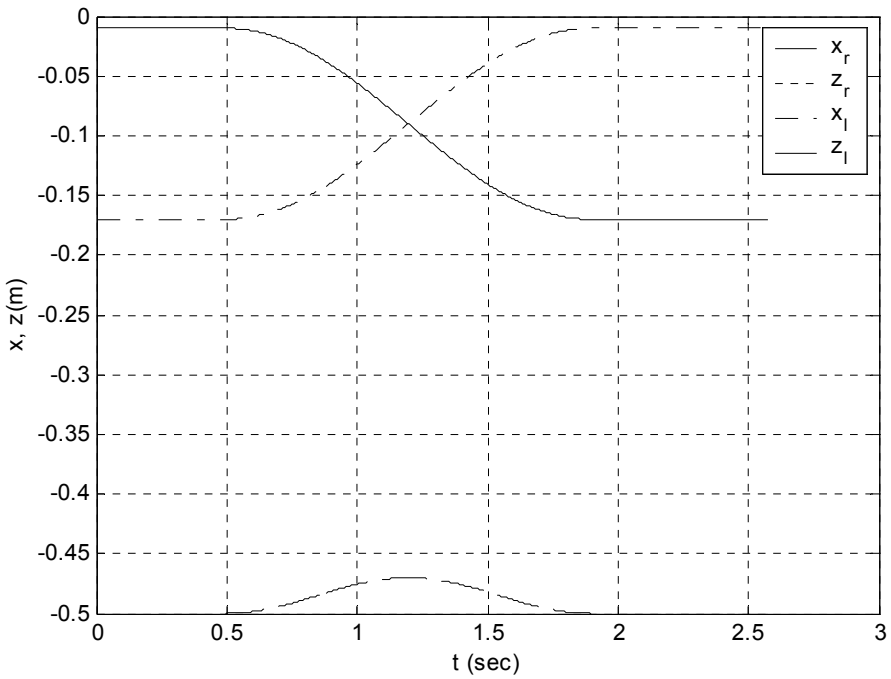

Figure 4.14. *x* and *z* trajectory for a reflex on applied at 1.9725 s for a load added at 1.9 s employing (4.3) and (4.4)

Figure 4.14. shows *x* and *z* trajectory for this reflex-on problem applied at 1.9725 s for a load added at 1.9 s using (4.3) and (4.4). It is noticed that there are no values added to the *x*-trajectory. Figure 4.15. displays the stability parameter β for both cases. It is inferred that no addition to the *x* trajectory in the double support phase enhances stability.

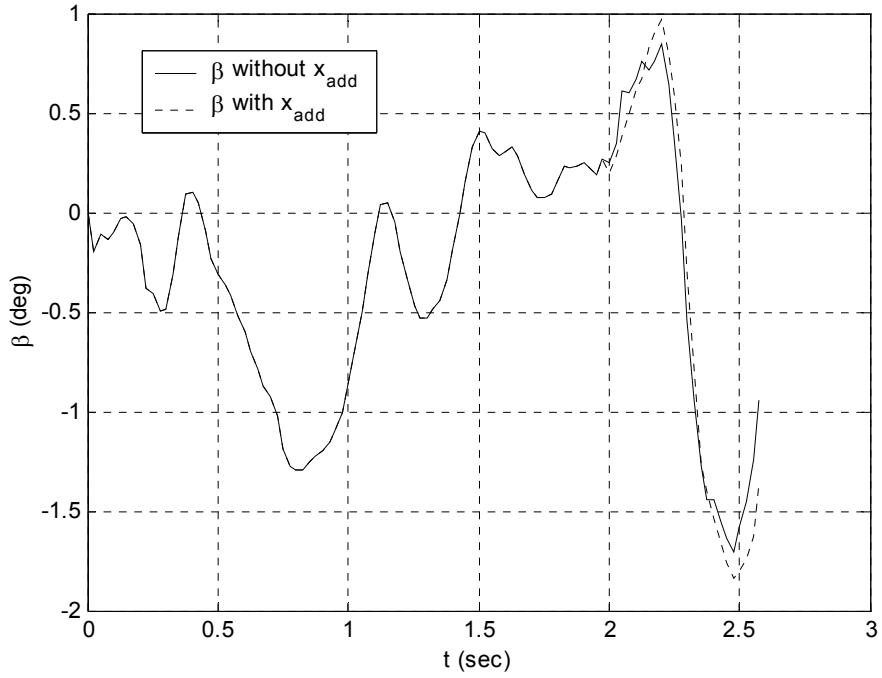

Figure 4.15. β vs. time for the reflexes with and without x_{refl}.

4.4. Modification on the Reflex-Adaptation Technique

The results in Section 4.2 recommend an automatic on-line adaptation of the values added to the x trajectories during the reflex period. Equations (4.1)-(4.4) can be merged together and applied to the algorithm as follows

$$x_{refl\_r} = \begin{cases} 0.01 & if\ Z_l > -h_b \\ -0.01 & if\ Z_r > -h_b \\ 0 & if\ Z_r = Z_l = -h_b \end{cases} \tag{4.5}$$

$$x_{refl\_l} = \begin{cases} -0.01 & if\ z\_l > -h_b \\ 0.01 & if\ z\_r > -h_b \\ 0 & if\ z\_r = z\_l = -h_b \end{cases} \tag{4.6}$$

Also Section 3.2 supposes an increase in the settling time. This leads to a better read of the filtered ZMP_x value to be used in shifting the center of the body. A new value of 4 s is set to the settling time.

Table 4.3. Experiments conducted with modified reflex-adaptation on. 1 indicates successful experiment and 0 indicates failure.

Load (kg) / Time (s)	2.5	5.0	7.5	10	12.5	15
0.1	1	1	1	1	1	1
0.2	1	1	1	1	1	1
0.3	1	1	1	1	1	1
0.4	1	1	1	1	1	1
0.5	1	1	1	1	1	1
0.6	1	1	1	1	1	1
0.7	1	1	1	1	1	1
0.8	1	1	1	1	1	1
0.9	1	1	1	1	1	1
1.0	1	1	1	1	1	1
1.1	1	1	1	1	1	1
1.2	1	1	1	1	1	1
1.3	1	1	1	1	1	0
1.4	1	1	1	1	1	0
1.5	1	1	1	1	1	1
1.6	1	1	1	1	1	1
1.7	1	1	1	1	1	1
1.8	1	1	1	1	1	1
1.9	1	1	1	1	1	1
2.0	1	1	1	1	1	1
2.1	1	1	1	1	1	1
2.2	1	1	1	1	1	1
2.3	1	1	1	1	1	1
2.4	1	1	1	1	1	1
2.5	1	1	1	1	1	1
2.6	1	1	1	1	1	1
2.7	1	1	1	1	1	1
2.8	1	1	1	1	1	1
2.9	1	1	1	1	1	1
3.0	1	1	1	1	1	1
3.1	1	1	1	1	1	1
3.2	1	1	1	1	1	1
3.3	1	1	1	1	1	1
3.4	1	1	1	1	1	1
3.5	1	1	1	1	1	1
3.6	1	1	1	1	1	1
3.7	1	1	1	1	1	0
3.8	1	1	1	1	1	0
3.9	1	1	1	1	1	1
4.0	1	1	1	1	1	1
4.1	1	1	1	1	1	1
4.2	1	1	1	1	1	1
4.3	1	1	1	1	1	1
4.4	1	1	1	1	1	1
4.5	1	1	1	1	1	1
4.6	1	1	1	1	1	1
4.7	1	1	1	1	1	1
4.8	1	1	1	1	1	1

CHAPTER 5

CONCLUSION

In this thesis, the problem of creating reflexes for a sudden addition of payloads given only the information that the payload has been added is considered. Responses that try to pull the body back by shifting the foot in touch with the ground forward with respect to the body shows better results than the ones trying to push the body forward. The later responses have the ability to lessen the tendency of falling forward because of attaching the load to the front side of the biped trunk.

The load addition simulations focus on cases where the reflex is triggered in the stance position (two feet are in touch with the ground). No shift of the feet is allowed. This allows a more effective and stable state after the reflex.

Adaptation of the trajectory consists of two parts: Shifting the *ZMP* (which is equal to the center of gravity when the robot is stationary), and generating a trajectory to retain the original step size using a small interpolation function. The results show that this gives a viable and stable recovery and prepares for a normal original walk.

Load addition reflexes are fundamental to many industrial applications. Load addition reflexive responses will facilitate the successful and viable walk of the biped in industrial environments with other reflexive strategies as those for foothold errors such as slipping on slippery surfaces and tripping on obstacles.

APPENDIX

CODE FUNCTIONS USED IN THE REFLEX-ADAPTATION TECHNIQUE

```
void controller()
{
    Matrix result;


    //INPUT VARIABLES
    velocity_vector=vvector;
    position_vector=xvector;
    pb=position_vector[mslice(0,0,3,1)];;
    wAblist=position_vector[mslice(3,0,9,1)];
    q=position_vector[mslice(12,0,2*n,1)];
    vb=velocity_vector[mslice(0,0,3,1)];
    wb=velocity_vector[mslice(3,0,3,1)];
    qdot=velocity_vector[mslice(6,0,(6+2*n)-7+1,1)];
    double
VV0[]={wAblist[0][0],wAblist[3][0],wAblist[6][0],wAblist[1][0],
    wAblist[4][0],wAblist[7][0],wAblist[2][0],wAblist[5][0],wAblist[
8][0]};
    wAb=Matrix(3,3,VV0);

    //joint actual variables
    qR=q[mslice(0,0,n,1)];
    qL=q[mslice(n,0,n,1)];
    qRdot=qdot[mslice(0,0,n,1)];
    qLdot=qdot[mslice(n,0,n,1)];

    //joint error variables
    qRerror=0.0;
    qRdoterror=0.0;
    qLerror=0.0;
    qLdoterror=0.0;

    referencecycle=referencecycle+1;

    if(
referencecycle>=2*(double_support_period+single_support_period)/steptime)
    {
       referencecycle=0;
    }

    if (TRIGGER_1 == 1)
    {
       referencecycle_reflex=referencecycle_reflex+1;
       Matrix PID_control_R(6,1),PID_control_L(6,1);
       //joint pos, vel references
       qref=refgencartesian_analytic_reflex(qR,qL);
       qRref=qref[mslice(0,0,n,1)];
```

```
            qLref=qref[mslice(n,0,n,1)];
            qRdotref=qref[mslice(2*n,0,n,1)];
            qLdotref=qref[mslice(3*n,0,n,1)];


            //p d i errors
            qRerror=qRref-qR;
            qLerror=qLref-qL;

            qRdoterror=qRdotref-qRdot;
            qLdoterror=qLdotref-qLdot;

            qRinterror=qRinterror+steptime*qRerror;
            qLinterror=qLinterror+steptime*qLerror;


            PID_control_R = Kp * qRerror + Kd * qRdoterror + Ki *
qRinterror;
            PID_control_L = Kp * qLerror + Kd * qLdoterror + Ki *
qLinterror;

            double
UU1[]={0,0,0,0,0,0,PID_control_R[0][0],PID_control_R[1][0],PID_control_R[
2][0],

         PID_control_R[3][0],PID_control_R[4][0],PID_control_R[5][0],PID_
control_L[0][0],

         PID_control_L[1][0],PID_control_L[2][0],PID_control_L[3][0],PID_
control_L[4][0],
            PID_control_L[5][0]};
            result= Matrix(18,1,UU1);
            uvector=result;

            }
         else
         {
            Matrix PID_control_R(6,1),PID_control_L(6,1);
            //joint pos, vel references
            qref=refgencartesian_analytic(qR,qL);
            qRref=qref[mslice(0,0,n,1)];
            qLref=qref[mslice(n,0,n,1)];
            qRdotref=qref[mslice(2*n,0,n,1)];
            qLdotref=qref[mslice(3*n,0,n,1)];


            //p d i errors
            qRerror=qRref-qR;
            qLerror=qLref-qL;

            qRdoterror=qRdotref-qRdot;
            qLdoterror=qLdotref-qLdot;

            qRinterror=qRinterror+steptime*qRerror;
            qLinterror=qLinterror+steptime*qLerror;
```

```
            PID_control_R  =  Kp  *  qRerror  +  Kd  *  qRdoterror  +  Ki  *
qRinterror;
            PID_control_L  =  Kp  *  qLerror  +  Kd  *  qLdoterror  +  Ki  *
qLinterror;

            double
UU1[]={0,0,0,0,0,0,PID_control_R[0][0],PID_control_R[1][0],PID_control_R[
2][0],

         PID_control_R[3][0],PID_control_R[4][0],PID_control_R[5][0],PID_
control_L[0][0],

         PID_control_L[1][0],PID_control_L[2][0],PID_control_L[3][0],PID_
control_L[4][0],
            PID_control_L[5][0]};
            result= Matrix(18,1,UU1);
            uvector=result;
         //output cartesian reference generation

         }


     }
```

```
Matrix refgencartesian analytic reflex(Matrix qR, Matrix qL)
{

    Matrix result;

    if (referencecycle_reflex == 1)
    {
       cout<<"reflex is applied at "<<tt<<endl;
       xr_s=rxgen(referencecycle);
       xl_s=lxgen(referencecycle);
       yr_s=rygen(referencecycle);
       yl_s=lygen(referencecycle);
       zr_s=rzgen(referencecycle);
       zl_s=lzgen(referencecycle);
       xdotr_s=rightxvel;
       xdotl_s=-rightxvel;
       ydotr_s=rightyvel;
       ydotl_s=leftyvel;
       zdotr_s=rightzvel;
       zdotl_s=leftzvel;

       xr_e=xr_s+x_reflex_add_r;
       xl_e=xl_s+x_reflex_add_l;
       yr_e=yr_s+y_reflex_add_r;
       yl_e=yl_s+y_reflex_add_l;
       zr_e=zr_s+z_reflex_add_r;
       zl_e=zl_s+z_reflex_add_l;

       diff=absl(xr_e-xl_e)/2.0;
       if(xr_e>=xl_e)
             factor=absl(xref_asymmetry+stepsize-xr_e)/(2*stepsize);

       else
             factor=absl(xref_asymmetry+stepsize-xl_e)/(2*stepsize);

             tss_adapt=single_support_period*acos(factor)/pi;

    tds_adapt=(double_support_period/single_support_period)*tss_adap
t;


    }

    double   UU1[]={0,0,   1,   RaddX+rxgen1(referencecycle_reflex)-
Ro_pSTAR_Ro[0][0],
          0,      1,      0,      RaddY+rygen1(referencecycle_reflex)-
Ro_pSTAR_Ro[1][0],
         -1,      0,      0,      RaddZ+rzgen1(referencecycle_reflex)-
Ro_pSTAR_Ro[2][0],
          0, 0, 0, 1};
    TRight=Matrix(4,4,UU1);
    dummyrefR=i_kine_analytic(TRight);


    dummydotrefR=(dummyrefR-dummyrefR_old)/steptime;
    dummyrefR_old=dummyrefR;
```

```
    //-----------------------------------------------------------------
---------//
         double  UU2[]={0,   0,   1,   LaddX+lxgen1(referencecycle_reflex)-
Lo_pSTAR_Lo[0][0],
             0,      1,      0,      LaddY+lygen1(referencecycle_reflex)-
Lo_pSTAR_Lo[1][0],
            -1,      0,      0,      LaddZ+lzgen1(referencecycle_reflex)-
Lo_pSTAR_Lo[2][0],
             0, 0, 0, 1};
         TLeft =Matrix(4,4,UU2);
         dummyrefL=i_kine_analytic(TLeft);

         dummydotrefL=(dummyrefL-dummyrefL_old)/steptime;
         dummyrefL_old=dummyrefL;

         double
UU3[]={dummyrefR[0][0],dummyrefR[0][1],dummyrefR[0][2],dummyrefR[0][3],
         dummyrefR[0][4],dummyrefR[0][5],dummyrefL[0][0],dummyrefL[0][1],
dummyrefL[0][2],
         dummyrefL[0][3],dummyrefL[0][4],dummyrefL[0][5],dummydotrefR[0][
0],dummydotrefR[0][1],
         dummydotrefR[0][2],dummydotrefR[0][3],dummydotrefR[0][4],dummydo
trefR[0][5],
         dummydotrefL[0][0],dummydotrefL[0][1],dummydotrefL[0][2],dummydo
trefL[0][3],
         dummydotrefL[0][4],dummydotrefL[0][5]};
         result=Matrix(24,1,UU3);
         return result;

    }
```

```
     double rxgen1(int refcycle)
     {

          double t,rightxpos,result;

          t = steptime*refcycle;

          if(t<=t_reflex_period)
          {
             rightxpos = xr_s + x_reflex_add_r /t_reflex_period*t;
          }
          if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+tds_adapt/2)) )
          {
             rightxpos=xr_e;


          }
          if      ((t_reflex_period+t_settling_time+tds_adapt/2)<t      &&
t<=(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry) )
          {
             if(once == 0)
             {
                   xref_asymmetry_add=  ZMPx_lp_filter - (-0.035) ;
                   xref_asymmetry=xref_asymmetry+xref_asymmetry_add;
                   once = 1;
             }
             rightxpos=xr_e;
             xr_e                  =                  xr_e                 +
xref_asymmetry_add/(2*t_x_ref_asymmetry/steptime);
          }

          if(
((t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)<t)     &&
(t<=
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)
) )
          {
             if(xr_e<=xl_e)
             {
                   rightxpos=(xref_asymmetry-diff)-(stepsize-diff)/2*(1-
cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*pi/tss_a
dapt ));
             }
             else
             {
                   rightxpos=(xref_asymmetry+diff)+(stepsize-diff)/2*(1-
cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*pi/tss_a
dapt ));
             }

          }
          if               (               (t_reflex_period               +
t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)<t )
          {
```

```
            if(xr_e<=xl_e)
            {
                  rightxpos=xref_asymmetry-stepsize;
                  if(                t>(t_reflex_period                +
1.5*t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt) )
                  {

         referencecycle=(single_support_period+double_support_period)/ste
ptime;
                        TRIGGER_1=2;
                  }
            }
            else
            {
                  rightxpos=xref_asymmetry+stepsize;
                  if(                t>(t_reflex_period                +
1.5*t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt) )
                  {
                        referencecycle=0;
                        TRIGGER_1=2;
                  }
            }
         }

         result=rightxpos;
         return result;

     }
```

```
     double rygen1(int refcycle)
     {
          double t,rightypos,result;
          if (refcycle==1)
          {
             y_reflex_add_r=-swing_offset-yr_s;
             yr_e=yr_s+y_reflex_add_r;
          }
          t = steptime*refcycle;
          if(t<=t_reflex_period)
          {
             rightypos=yr_s+y_reflex_add_r/t_reflex_period*t;
          }
          if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+t_x_ref_asymmetry)) )
          {
             rightypos=yr_e;
          }
          if(   ((t_reflex_period+t_settling_time+t_x_ref_asymmetry)<t)   &&
(t<=(t_reflex_period+t_settling_time+t_x_ref_asymmetry+tds_adapt+tss_adap
t)) )
          {
             if(xr_e<=xl_e)
                   rightypos=yref_assymetry-
swing_offset+swing*0.5*factor*(1-cos(2*(t-
(t_reflex_period+t_settling_time+t_x_ref_asymmetry))*pi/
                   (tds_adapt+tss_adapt)));
             else
                   rightypos=yref_assymetry-swing_offset-
swing*0.5*factor*(1-cos(2*(t-
(t_reflex_period+t_settling_time+t_x_ref_asymmetry))*pi/
                   (tds_adapt+tss_adapt)));
          }
          if               (               (t_reflex_period               +
t_settling_time+t_x_ref_asymmetry+tds_adapt+tss_adapt)<t )
          {
             rightypos=yref_assymetry-swing_offset;
          }

          result=rightypos;
          return result;

     }
```

```
     double rzgen1(int refcycle)
     {
          double t,rightzpos,result;
          if (refcycle==1)
          {
             z_reflex_add_r=-body_height-zr_s;
             zr_e=zr_s+z_reflex_add_r;
          }
          t = steptime*refcycle;

          if(t<=t_reflex_period)
          {
             rightzpos=zr_s+z_reflex_add_r/t_reflex_period*t;
          }
          if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)) )
          {
             rightzpos=zr_e;
          }
          if(
((t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)<t)     &&
(t<=
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)
) )
          {
             if(xr_e<=xl_e)
             {
                   rightzpos=zr_e;
             }
             else
             {
                   rightzpos=-body_height+0.5*step_height*factor*(1-
cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*2*pi/tss
_adapt));
             }


          }
             if              (              (t_reflex_period              +
t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)<t )
          {
             rightzpos=zr_e;

          }

          result=rightzpos;
          return result;

     }
```

```
double lxgen1(int refcycle)
{

    double t,leftxpos,result;

    t = steptime*refcycle;


    if(t<=t_reflex_period)
    {
       leftxpos = xl_s + x_reflex_add_l /t_reflex_period*t;
    }
    if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+tds_adapt/2+t_reflex_period)) )
    {
       leftxpos=xl_e;
    }
    if      ((t_reflex_period+t_settling_time+tds_adapt/2)<t      &&
t<=(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry) )
    {

       leftxpos=xl_e;
       xl_e                =                 xl_e                +
xref_asymmetry_add/(2*t_x_ref_asymmetry/steptime);
    }

    if(
((t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)<t)     &&
(t<=
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)
) )
    {
       if(xl_e<=xr_e)
       {
             leftxpos=(xref_asymmetry-diff)-(stepsize-diff)/2*(1-
cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*pi/tss_a
dapt ));
       }
       else
       {
             leftxpos=(xref_asymmetry+diff)+(stepsize-diff)/2*(1-
cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*pi/tss_a
dapt ));
       }

    }
    if               (               (t_reflex_period               +
t_settling_time+tds_adapt/2+tss_adapt+t_x_ref_asymmetry)<t )
    {
       if(xr_e<=xl_e)
             leftxpos=xref_asymmetry+stepsize;
       else
             leftxpos=xref_asymmetry-stepsize;

    }
```

```
    result=leftxpos;
    return result;

}
```

```
    double lygen1(int refcycle)
    {
        double t,leftypos,result;

        if (refcycle==1)
        {
           y_reflex_add_l=swing_offset-yl_s;
           yl_e=yl_s+y_reflex_add_l;
        }
        t = steptime*refcycle;
        if(t<=t_reflex_period)
        {
           leftypos=yl_s+y_reflex_add_l/t_reflex_period*t;
        }
        if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+t_x_ref_asymmetry)) )
        {
           leftypos=yl_e;
        }
        if(  ((t_reflex_period+t_settling_time+t_x_ref_asymmetry)<t)   &&
(t<=(t_reflex_period+t_settling_time+t_x_ref_asymmetry+tds_adapt+tss_adap
t)) )
        {
           if(xr_e<=xl_e)

        leftypos=yref_assymetry+swing_offset+swing*factor*0.5*(1-
cos(2*(t-(t_reflex_period+t_settling_time+t_x_ref_asymmetry))*pi/
                 (tds_adapt+tss_adapt)));
           else
                 leftypos=yref_assymetry+swing_offset-
swing*factor*0.5*(1-cos(2*(t-
(t_reflex_period+t_settling_time+t_x_ref_asymmetry))*pi/
                 (tds_adapt+tss_adapt)));
        }
        if               (               (t_reflex_period               +
t_settling_time+t_x_ref_asymmetry+tds_adapt+tss_adapt)<t )
        {
           leftypos=yref_assymetry+swing_offset;
        }
        result=leftypos;
        return result;
    }
```

```
     double lzgen1(int refcycle)
     {

          double t,leftzpos,result;

          if (refcycle==1)
          {
             z_reflex_add_l=-body_height-zl_s;
             zl_e=zl_s+z_reflex_add_l;
          }
          t = steptime*refcycle;


          if(t<=t_reflex_period)
          {
             leftzpos=zl_s+z_reflex_add_l/t_reflex_period*t;
          }
          if(                    (t_reflex_period<t)                    &&
(t<=(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)) )
          {
             leftzpos=zr_e;
          }
          if(
((t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry)<t)     &&
(t<=
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)
) )
          {
             if(xr_e<=xl_e)
             {
                   leftzpos=-body_height+0.5*step_height*factor*(1-cos((t-
(t_reflex_period+t_settling_time+tds_adapt/2+t_x_ref_asymmetry))*2*pi/tss
_adapt));
             }
             else
             {
                   leftzpos=zl_e;
             }

          }
          if               (               (t_reflex_period               +
t_settling_time+tds_adapt/2+t_x_ref_asymmetry+tss_adapt)<t )
          {
             leftzpos=zl_e;

          }


          result=leftzpos;
          return result;

     }
```

REFERENCES

[1] Forssberg, H., "Stumbling correct reaction: A phase-dependent compensatory reaction during locomotion," Journal of Neuro-physiology, vol. 42, pp. 936-953,1979.

[2] Forssberg, H., S. Grillner, S. Rossignol, and P. Wallen, "Phasic Control of reflexes during locomotion in vertebrates," Neural Control of Locomotion, vol. 18 of Advances in Behavioral Biology, pp.647-674.1976.

[3] Belanger, M., and A.E.Patla, "Corrective responses to perturbation applied during walking in humans," Neuroscience Letters, vol. 49, pp. 291-295, 1984.

[4] Tomovic, R., and G. Boni,"An adaptive artificial hand," IRE Translations on Automatic Control, vol. AC-7, pp. 3-10, 1962.

[5] Bekey, G. A., and R. Tomovic, "Robot Control by reflex actions," in Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, 1986.

[6] Wong, H.C., and D.E. Orin, "Reflex Control of the Prototype Leg during Contact and Slippage," in proceedings of the IEEE international Conference on Robotics and Automation, Philadelphia, PA, 1988.

[7] Hirose, S.,"A study of design and control of quadruped walking vehicle," International journal of robotics Research, vol. 3, pp.113-133, 1984.

[8] Brooks, R. A., "A robot that walks: Emergent behaviors from a carefully evolved network," in Proceedings of the IEEE International Conference on Robotics and Automation,Scottsdale,AZ,1989.

[9] Weng, S., and K. Young.,"Robot impact control inspired by human reflex," in proceedings of the IEEE International Conference on Robotics and Automation, pp. 2579-2583, Minneapolis, MN, 1996.

[10] Boone, G. N.and J. K. Hodgins, "Slipping and Tripping Reflexes for Bipedal Robots," Technical report, Georgia Institute of Technology, Atlanta.

[11] Park, J. H. and O. Kwon, "Reflex Control of Biped Robot Locomotion on a Slippery Surface", Proc. IEEE Int. Conf. Robotics and Automation, ICRA 2001, pp. 4134-4139, Seoul, Korea, 2001.

[12] M. H. Railbert, "Legged robots that balance," MIT Press, Cambridge, 1986.

[13] Hodgins, J. K., and M. H. Raibert, "Running experiments with a planar biped," in O. Faugerasand G. Giralt, editors, Robotics Research : The Third International Symposium, Cambridge, 1986. MIT Press.

[14] Jong H. Park, Ohung Kwon, "Reflex Control of Biped Robots to Prevent Foot Slips During Locomotion on A Slippery Floor," Proceedings of International Workshop on Bio-Robot and Teleoperation (IWBRT2001), pp. 46-53, Beijing, China, May 20024.

[15] Hodgins, J., and M.H. Raibert, "Adjusting Step Length for Rough Terrain Locomotion ," IEEE Transactions on Robotics and Automation, vol. 7, pp.289-298, 1991.

[16] Hiroaki FUNABASHI, Yukio TAKEDA, Shigenari ITOH and Masaru HIGUCHI : Disturbance Detection and Compensation of a Biped Walking Machine Based on Reflex Motions, Proc.2nd International Conference on Climbing and Walking Robots, Portsmouth, UK, 13-15 Sept 1999, pp547-558.

[17] Lewis, M. A. , and L. S. Simo, "Certain Principles of Biomorphic Robots," International Journal of Robotics Reserch, 1999. In Press

[18] Kimura, H., Y. Fukuoka, Y. Hada, and K. Takase. 3D Adaptive Dynamic Walking of a quadruped Robot by Using Neural System model. in Proc. of 4th Int. Conf. on Climbing and Walking Robots (CLAWAR2001). 2001. Karlsruhe.

[19] Kajita S., and K. Tani, "Adaptive Gait Control of a Biped Robot based on Realtime Sensing .of the Ground Profile," *Proc. lEEE lnt. Conf. on Robotics and Automatıon,* pp. 570-577 ,1996.

[20] Park, J. H., and H. C. Cho, "An On-Line Trajectory Modifier for the Base Link of Biped Robots To Enhance Locomotion Stability," Proceedings of the 2000 IEEE

International Conference on Robotics & Automation San Francisco, CA • April 2000 pp. 3353-3358

[21] W. T. Miller, A. L. Kun, "Dynamic Balance of a Biped Walking Robot," in O. Omidvar, and P. van der Smagt, editors, "Neural Systems for Robotics," pp. 17-35, Academic Press, San Diego, 1997.

[22] Anthony, M. L., and L. S. Simo A Model of Visually Triggered Gait Adaptation," International Symposium on Adaptive Motion of Animals and Machines (AMAM) Montreal, Canada, August 8-11, 2000

[23] Jan Albiez, Tobias Luksch, Winfried Ilg, Karsten Berns "Reactive Reflex based Posture Control for a Four-Legged Walking Machine", CLAWAR 2001, 24. - 26. Sept. 2001, Karlsruhe, Germany

[24] Papadopoulos, D. and M. Buehler," Stable Running in a Quadruped Robot with Compliant Legs," Center for Intelligent Machines, Ambulatory Robotics Laboratory. IEEE Int. Conf. Robotics and Automation, San Francisco, 2000

[25] Luh, J. Y. S., M. W. Walker and R. P.C. Paul, "On-Line Computational Scheme for Mechanical Manipulators," Jour. Dynamic Systems, Measurement, and Control, Vol. 102, pp. 69-76, June 1980.

[26] Fujimoto, Y. and A. Kawamura, "Simulation of an Autonomous Biped Walking Robot including Environmental Force Interaction," IEEE Robotics and Automation Magazine, pp. 33-42, June 1998