ARTICLE TEMPLATE

Humanoid walking pattern generation based on model predictive control approximated with basis functions

Ang Zhang[a] Ixchel G. Ramirez-Alpizar[a], Kévin Giraud Esclasse[b], Olivier Stasse[b] and Kensuke Harada[a]

[a]Department of Systems Innovation, Graduate School of Engineering Science, Osaka University, 1-3 Machikaneyama-cho, Toyonaka, Osaka, 560-8531, Japan; [b]CNRS, LAAS, Université de Toulouse, 7 avennue du Colonel Roche, F-31400, Toulouse, France

ARTICLE HISTORY
Compiled March 4, 2019

ABSTRACT
This paper proposes a real-time walking pattern generator (WPG) based on model predictive control (MPC). Since reducing the calculation time is a crucial problem in real-time WPG, we consider introducing basis functions to reduce the number of control input. The control inputs in the MPC are described by a series of basis functions. Compared with the standard discrete-time MPC formulation, the approach with basis functions requires fewer optimization variables at the cost of decreasing precision. In order to find an appropriate trade-off, two basis functions named Laguerre functions and Haar functions, are tested in this paper. MPC with Laguerre functions decreases more computational load while MPC with Haar functions offers a more accurate solution. The approach is not restricted to Laguerre functions or Haar functions, users can select their own basis functions for different applications and preferences.

KEYWORDS
Humanoid robot; bipedal locomotion; linear model predictive control; Laguerre orthonormal functions; Haar functions

1. Introduction

Humanoid robots are promising candidates for a broad variety of applications, for example, disaster relief, industrial labor and domestic assistance [1]. Advanced motion controllers are now closer to realtime feasibility, but they require powerful multi-core CPUs. Specially, the bipedal nature of humanoids requires footstep planning which is computationally expensive and in realtime cases, the sensor information should be managed in the meantime. Therefore, to improve humanoids' locomotion, it is important to have a fast online walking pattern generator(WPG).

In realtime walking motion generation, Linear Inverted Pendulum Model (LIPM) has been used in many studies [2–4]. By using LIPM, an extension of linear quadratic regulator control by preview control is used for a closed loop approach to address the problem of bipedal walking pattern generation [5]. Harada et al.[6] generated a realtime biped gait using an analytical solution. However, these methods calculate the

CONTACT Ang Zhang. Email: zhang@hlab.sys.es.osaka-u.ac.jp

walking pattern one walking step in advance. Morisawa et al. [7] proposed an analytical solution based WPG which can maintain balance against disturbance. Nishiwaki et al. [8] proposed a preview control based realtime WPG which calculation cycle is about 10 to 40ms. A method of combining Gauss Pseudospectral with preview control based WPG is tested on HRP-2 [9].

Englsberger et al. [10] proposed the control of the unstable dynamics of capture point (CP) and extended the CP to a 3 dimensional divergent component of motion (DCM) [11]. However, it is difficult to add constraints for the center of pressure (CoP) in their design. The utilization of MPC allows the incorporation of CoP constraints in the controller design [12]. Romualdi et al. [13] compared the walking performance of DCM based on MPC with that of DCM based on instantaneous controller [10]. The CoP trajectory generated by MPC is smoother while the instantaneous controller achieved a faster walking velocity.

There are a lot of attempts in WPG including several constraint conditions such as CoP. For realizing such purposes, MPC based WPG was proposed and has been proven to be successful [14,15]. It has also been extended to have abilities such as obstacle avoidance [16] and disturbance rejection [17].

Reducing the calculation time in realtime WPG is important, since there are computational loads from analysis of sensor data. Few attempts have been tried on reducing the calculation time of MPC based realtime walking pattern generator. In order to have a faster online WPG, we go further and propose reformulating the control signals of the MPC using basis functions, as illustrated in Figure 1.

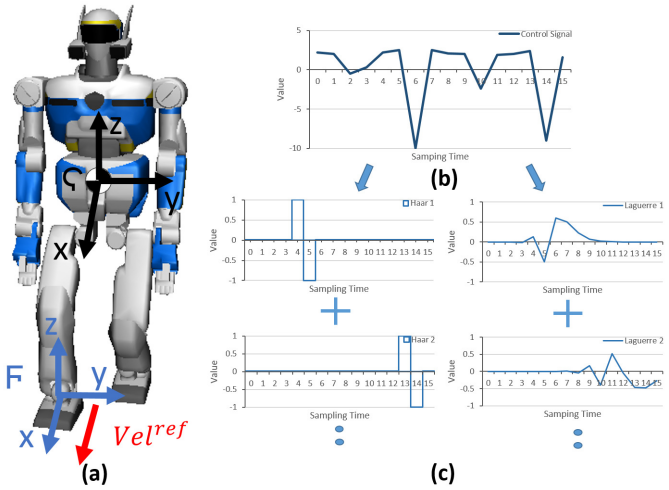


Figure 1. In (a), the figure shows two frames ς, F, representing CoM and the supporting foot respectively. Figure (b) shows the control input in MPC based WPG and two extreme values. In order to improve computational efficiency, we use Haar(left part in (c)) or Laguerre(right part in (c)) basis functions to reformulate the control signal.

Decreasing the number of control inputs is a solution to improve efficiency in the MPC [18,19]. In the conventional discrete time MPC [14,15], the control signal can be considered as a combination of pulse basis functions. To cover the whole time horizon(N_p), the number of basis functions required is N_p. In this paper, we propose applying Haar and Laguerre basis functions to approximate the control signal. Since Haar and Laguerre functions carry more information than pulse functions, fewer basis functions are needed for representing the input, leading to a reduction in the number of

control inputs. Furthermore, Haar and Laguerre functions specially suit the property of inputs in WPG. The control signal in Figure 1 might be difficult to approximate due to their sharp changes. However, Haar functions can naturally represent the lows while Laguerre functions are embedded with a tuning parameter which modifies them to fit the desired signal.

Basis functions are not new and have been already treated in several works [20–23], where the control signal is represented by a set of basis functions, for example, Laguerre, Kauz and Haar. It provides a more compact problem description with fewer control variables, which leads to computational benefits. Haar functions are used to build a transformation matrix which changes the signal from time domain into wavelet domain. This transformation decomposes signals into several different frequency components without destroying their time domain structures [24]. Laguerre functions have been applied to the identification of linear time invariant systems for a long time [25]. In system identification, the purpose of implementing Laguerre functions is to get low order models and to incorporate a priori information to the system's time constants.

The main contribution of this paper is a basis function approach to the MPC based WPG. The method is a parametric description of the MPC's control inputs via linear combinations of basis functions. Moreover, instead of restricting to special types of basis functions, we provide a general parametrization approach which allows users to select their own basis function. To fit the property of bipedal walking, Haar and Laguerre functions are introduced. Examples of two MPC designed with these different basis functions are shown and compared.

This paper is organized as follows: Section 2 introduces the dynamic walking motion for both the bipedal's body and feet. Section 3 discusses the MPC scheme. Cost function and constraints are discussed in this part. Then, the properties of walking motion are analysed in Section 4. The Laguerre and Haar basis functions and its use are proposed in Section 5. Finally the conclusion and future work are introduced.

2. Dynamic Walking Motion

2.1. Motion Model of Center of Mass (CoM)

In Figure 1, a frame S is attached to the position of the CoM and to the orientation of the robot's trunk. The orientation θ is around vertical axis Z in the frame. In order to generate a smooth motion of the CoM, we consider that its trajectory is differentiable three times. The jerks are assumed to be piece-wise linear on the time intervals $T = 0.1s$. We define the superscript C which implies the states on the CoM. $C \in \{c_x, c_y\}$ indicates the axes of the horizontal plane in the frame S, the future states $\hat{x}^C_{k+1} = \begin{bmatrix} x^C_{k+1} & \dot{x}^C_{k+1} & \ddot{x}^C_{k+1} \end{bmatrix}^{\mathrm{T}}$ (including the CoM's position, velocity and acceleration) can be generated by,

$$\hat{x}^C_{k+1} = A\hat{x}^C_k + B\dddot{x}^C_k, \tag{1}$$

where

$$A = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix},$$

Eq. (1) is the motion model of the CoM. The real motion of the CoM will be set to follow the motion model of the CoM. To define the CoM over the prediction horizon N_p, the vector X^C_{k+1} is used to express the positions of the CoM over the whole horizon, $X^C_{k+1} = \begin{bmatrix} x^C_{k+1} & \cdots & x^C_{k+N_p} \end{bmatrix}^{\mathrm{T}}$, similar definitions are used to describe $\dot{X}^C_{k+1}$, $\ddot{X}^C_{k+1}$ and $\dddot{X}^C_{k+1}$. Prediction horizon (N_p) indicates the length of state variables X^C_k.

By defining the state $\hat{X}^c_{k+1} = \begin{bmatrix} X^C_{k+1} & \dot{X}^C_{k+1} & \ddot{X}^C_{k+1} \end{bmatrix}^{\mathrm{T}}$, and the control input $U^C_k = \dddot{X}^C_k$ of the following state equation, we obtain the prediction model of the CoM,

$$\hat{X}^C_{k+1} = A_c \hat{X}^C_k + B_c U^C_k, \tag{2}$$

where matrices A_c, B_c can be obtained from the recursive application of Eq. (1).

To distinguish the prediction horizon N_p, the length of control input U^C_k is defined as control horizon N_c. This is because the control horizon can be reduced by basis functions while the prediction horizon is always kept with same value.

The orientation of the CoM, X^{C_θ} in frame $\mathcal{S}$ can be calculated with a similar prediction form of Eq. (2) (see [15] for more details).

2.2. Motion of the Center of Pressure (CoP)

Assuming that the centroidal dynamic [26] which is the dynamic of a humanoid robot projected at its CoM is linear (the angular momentum produced by the rotations of the robot is supposed to be zero and the CoM evolves on a horizontal plane [16]) when all the contacts with the environment are coplanar, the CoP should strictly lie in the support polygon in order to meet the balance criteria. The positions of CoP, y^C, can be approximated by LIPM. The CoP is predicted by a linear function of the CoM,

$$y^C_{k+1} = C \hat{x}^C_{k+1} \tag{3}$$

where $C = \begin{bmatrix} 1 & 0 & -h/g \end{bmatrix}$, h and g are the height of the CoM and the norm of the gravity vector, respectively.

The vector of the CoP over the prediction horizon is defined as $Y^C_{k+1} = \begin{bmatrix} y^C_{k+1} & \cdots & y^C_{k+N_p} \end{bmatrix}^{\mathrm{T}}$, thus we have

$$Y^C_{k+1} = C_c \hat{X}^C_{k+1} \tag{4}$$

where matrix C_c is a collection of C in Eq. (3).

2.3. Motion of Foot

Consider that a frame $\mathcal{F}$ is attached to the current support foot (see Figure 1) , in this case x^F_k with superscript F indicates the current state of the foot. $F \in \{f_x, f_y, f_\theta\}$ shows the position of the foot's center in X, Y direction and the angle between the foot's orientation and X axis. Position and orientation of the foot are shown in Figure 2. The future steps X^F_{k+1} representing the states of the foot at each time step are denoted by $X^F_{k+1} = \begin{bmatrix} x^F_{k+1} & \cdots & x^F_{k+N_p} \end{bmatrix}^{\mathrm{T}}$.

The future steps X_{k+1}^F can be calculated by the current step x_k^F and the control input U_k^F,

$$X_{k+1}^F = A_f x_k^F + B_f U_k^F, \tag{5}$$

where A_f, B_f are coefficient matrices indicating which step is active in the sampling interval (see [15] for details).

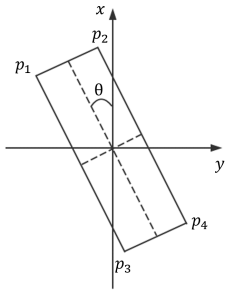


Figure 2. p_i on the edge of the feet indicates the support polygon and θ denotes orientation.

In order for the motion to be feasible and stable, the CoP is constrained to be inside the support polygon. The position of the feet is supposed to be directly below the ankles. To cope with Eq. (4), the positions of the feet Y_k^F are defined as,

$$Y_k^F = IX_k^F \tag{6}$$

where I is the identity matrix.

3. Model Predictive Control

3.1. Walking Pattern Generator

Walking motions can be produced by following a reference velocity. The reference velocity $Vel_{k+1}^{ref} = \begin{bmatrix} Vel_{k+1}^{x,ref} & Vel_{k+1}^{y,ref} & Vel_{k+1}^{\theta,ref} \end{bmatrix}^{\mathrm{T}}$ that describes velocities in X and Y directions and the yaw velocity of the waist in Figure 1, is provided to the WPG, which computes the foot steps and CoM jerks. Then, the CoM trajectory is determined from its jerks and the trajectories of feet are generated from the foot steps positions found by the MPC. In the next sampling time, the WPG is reinitialized with the current targeted velocity and with the updated states from the dynamic motion generator which transforms the given path into a dynamically executable robot trajectory.

3.2. Position Tracking

We define the control input for position control as $U_k^{x,y} = \begin{bmatrix} U_k^{c_x} & U_k^{f_x} & U_k^{c_y} & U_k^{f_y} \end{bmatrix}^{\mathrm{T}}$. $U_k^{c_x}$ and $U_k^{c_y}$ are from U_k^C in Eq. (2), with superscript $C \in \{c_x, c_y\}$ indicating the control input for the CoM. $U_k^{f_x}$ and $U_k^{f_y}$ are from U_k^F in Eq. (5) where superscript $F \in \{f_x, f_y, f_\theta\}$ indicates the control input of the foot.

To optimize the cost function related to position tracking, we defined

$$\begin{aligned}\min_{U_k^{x,y}} \frac{\alpha}{2}(||\dot{x}_{k+1}^{c_x} - Vel_{k+1}^{x,ref}||^2 + ||\dot{x}_{k+1}^{c_y} - Vel_{k+1}^{y,ref}||^2)+ \\ \frac{\beta}{2}(||Y_{k+1}^{f_x} - Y_{k+1}^{c_x}||^2 + ||Y_{k+1}^{f_y} - Y_{k+1}^{c_y}||^2)+ \\ \frac{\gamma}{2}(||\dddot{U}_{k+1}^{c_x}||^2 + ||\dddot{U}_{k+1}^{c_y}||^2)\end{aligned} \tag{7}$$

where α, β, γ are weights. The optimization problem firstly compares the linear velocity, secondly it generates the CoP ($Y_{k+1}^{c_x}$, $Y_{k+1}^{c_y}$) using Eq. (4) tracks the feet positions ($Y_{k+1}^{f_x}$, $Y_{k+1}^{f_y}$) generated by Eq. (6) which ensures balance, thirdly it minimizes the jerks in order to get a smooth motion of the robot.

The optimization problem of Eq. (7) can be expressed in a canonical form

$$\min_{U_k^{x,y}} \frac{1}{2} U_k^{x,yT} Q_k U_k^{x,y} + {p_k}^T U_k^{x,y}, \tag{8}$$

where the Q_k matrix and p_k matrix are as defined in [15]. The control horizon N_c is equal to the length of the control input $U_k^{x,y}$.

When solving this kind of Quadratic Programming(QP) problem (Eq. (8)), the length of the control horizon and the structures of Q_k, p_k have a significant influence on the computational load. In this paper, Haar and Laguerre basis functions will be implemented to approximate control inputs $U_k^{C_x}, U_k^{C_y}$ in $U_k^{x,y}$, so that a new QP problem, with control inputs of fewer dimensions, is built.

3.3. Orientation Tracking

Let us assume that the yaw motion of the robot's waist can be written using the same linear dynamic forms given in Eq. (2, 4) and define the control input for orientation as U_k^{θ}. As rotation of the robot's posture is related to both the waist and feet, here we consider that the orientation of the feet are aligned with the waist most of the time. We can determine the orientations of the feet by solving the following objective function,

$$\min_{U_k^{\theta}} \frac{\alpha}{2}||\dot{x}_{k+1}^{c_\theta} - Vel_{k+1}^{\theta,ref}||^2 + \frac{\beta}{2}||\sum_k (\dot{x}_{k+1}^{f_\theta} - Vel_{k+1}^{\theta,ref})||^2. \tag{9}$$

where $\dot{x}_{k+1}^{c_\theta}$ indicates the changing of yaw angle in the waist and $\dot{x}_{k+1}^{f_\theta}$ shows the changing of orientation in the foot. The objective function lets the waist trace the targeted orientation and also considers the angle between the waist and the feet. In the same way as the position controller, the input U_k^{θ} can also be represented by a set of basis functions.

3.4. Constraints

The balance of humanoids should be ensured and the feasibility of the foot step needs to be verified. Constraints are introduced to achieve a safe and feasible walking.

3.4.1. Balance Constraints

Constraints should be added to ensure balance while walking, which is determined by the CoP. To achieve a balanced bipedal walking, the CoP must stay in the support polygon. Only single support phase is considered and the CoP stays in the support polygon when it satisfies,

$$A_{cop,k}(U_k^\theta)U_k^{x,y} \leq \overline{U_{cop,k}} \tag{10}$$

where $\overline{U_{cop,k}}$ is the upper bound column vector. Eq.(10) indicates that the i-th element in the left vector is smaller than the i-th element in $\overline{U_{cop,k}}$. Detailed explanations of the CoP constraints can be found in [16]. $A_{cop}(U_k^\theta)$ is a matrix depending on U_k^θ which makes this constraint non-linear.

However, the non-linear issue can be avoided if U_k^θ is pre-calculated[15]. To keep the linear form of QP, the orientations problem will be calculated prior to the position controller based on Eq.(8).

3.4.2. Feasibility Constraints

It is also necessary to set constraints to ensure the movements of the feet computed by the WPG are feasible with respect to joint angles, velocity limitations, self-collision, over-stretching avoidance and kinematic limitations. The same convex hulls as in [16] are used to describe the feet limitations and the swing foot has to land inside the convex hull, i.e.

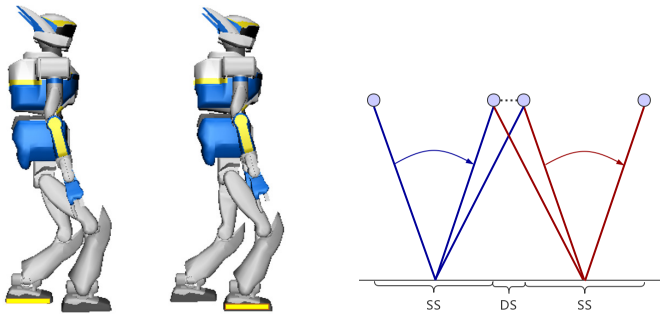
$$A_{foot,k}(U_k^\theta)U_k^{x,y} \leq \overline{U_{foot,k}}. \tag{11}$$

4. Properties of Control Signal

Unlike conventional MPC [14,15], we use basis functions to approximate control input U_k^C in Eq. (2). The physical meaning of control input are jerks of the CoM, $\dddot{X}_k^C$ which can be separated into two directions $\dddot{X}_k^{c_x}$, $\dddot{X}_k^{c_y}$ on X, Y axis. Before reformulation of the jerk, let us have a look at velocity and acceleration of the CoM.

4.1. Linear Inverted Pendulum Model

The WPG is based on a 3D LIPM. In Figure 3(b), the walking motion of the LIPM is separated into single support (SS) phase and double support (DS) period. With $N_p(= 16)$ time intervals (0.1s), the time horizon of prediction is 1.6s. The full duration of one step is 0.8s, including single support (0.7s) and double support (0.1s). Two steps are predicted in a prediction horizon. Figure 4 shows the velocity and acceleration of the robot in X direction during a horizon. Small fluctuations happen in SS at $0 \sim 0.2$s and 0.8s$\sim$1.0s, these are related to the dynamic of the LIPM. During the DS period ($t = 0.6$s and 1.4s), the acceleration jumps from maximum to minimum and the CoP moves from the backward support foot to the forward one, see Figure 3(b).



(a) The yellow marks indicate the support feet in SS before and after the DS period.

(b) A full step includes SS and DS.

Figure 3. During DS period, the CoP moves from one foot to the other.

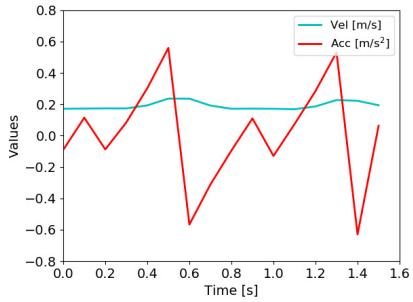


Figure 4. Detailed information about the velocity and acceleration of the robot in X direction during a prediction horizon. At the time periods 0.5s∼ 0.6s and 1.3s∼ 1.4s, the acceleration jumps from maximum to minimum.

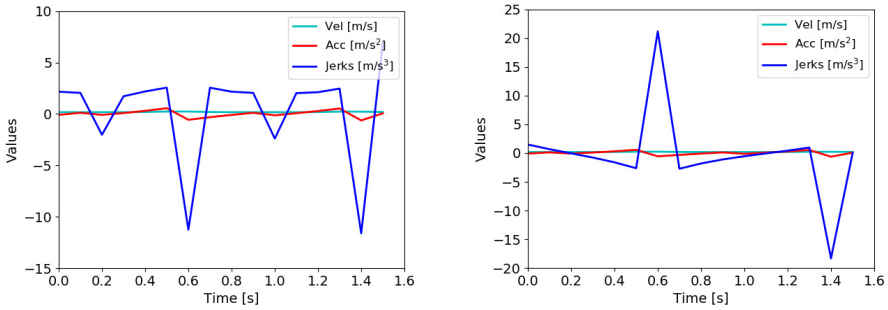
4.2. The jerks

The changes of acceleration in DS require a big jerk as shown in Figure 5. Unlike the conventional MPC, we use basis functions to approximate control inputs U_k^C which are the jerks of CoM, $\dddot{X}_k^C$. However, two deep lows in the jerk make the control input hard to be represented. To solve this problem, we introduce Haar functions and Laguerre functions which are able to represent this kind of signal.

As depicted in Figure 5, jerks along X and Y axis have peaks. The jerk along the Y axis is more difficult to catch as it changes sign due to the feet transition.

5. Approximation with Basis Functions

In the conventional discrete time MPC [14,15], the control input $\dddot{X}_k^C$ in Figure 5 is represented by a series of pulse basis functions. In pulse basis functions, as shown in Figure 6, information is carried at certain sampling time. In order to cover the whole time horizon($N_p = 16$), the number of pulse functions required is N_p. The length of the control input is the number of basis functions, which is $N_c = 16$. To reduce the number of basis functions, Haar basis functions and Laguerre basis functions are implemented to approximate the control input of the MPC.



(a) Along X direction, the jerk has two deep lows at $t = 0.6, 1.4$s.

(b) Along Y direction, the jerk has a peak and a low at $t = 0.6, 1.4$s.

Figure 5. The velocity, acceleration and jerks of CoM along X and Y directions. Extreme values of jerks happened in the DS phase.

There are two reasons for applying Laguerre and Haar functions. Firstly, both a single Laguerre function (in Figure 7) and a single Haar function (in Figure 8) carry more information compared with pulse functions. As a result, to represent the same control signal, fewer number of basis functions are needed which also means the dimension of the control input is reduced. Secondly, Haar and Laguerre functions specially suit the property of the WPG's control input. From Figure 5, it can be seen that two extreme values (at $t = 0.6s$ and $1.4s$) appear in the control input due to the LIPM and such a sharp change makes it hard for basis functions to approximate the control input. However, Haar functions can naturally represent the values and steep changes in inputs. Laguerre functions are embedded with a tuning parameter a which can modify the shape of Laguerre functions to fit the property of the desired signal. Details of Haar functions and Laguerre functions are discussed in this Section.

5.1. Input in Conventional MPC

Before introducing Laguerre functions, let us recall the control inputs of the MPC from Eq. (2) and express them in the time horizon.

$$U_k^C = \begin{bmatrix} u_k^C & u_{k+1}^C & \dots & u_{k+N_c-1}^C \end{bmatrix}^{\mathrm{T}} \tag{12}$$

The dimension of the control input is N_c which is the control horizon. At time k, elements with U_k^C can be represented by the discrete δ functions combined with U_k^C,

$$u_{k+i}^C = \begin{bmatrix} \delta_i & \delta_{i+1} & \dots & \delta_{i-N_c+1} \end{bmatrix} U_k^C, \tag{13}$$

where $\delta_i = 1$, if the subscript $i = 0$; $\delta_i = 0$ if $i \neq 0$. Here the δ functions are considered as a pulse operator and the δ_{i-d} shifts the pulse forward as index d increases. Therefore, the pulse functions (or step functions) are representing the control input if we take U_k^C as the coefficient vector. Other thing that should be noted is that the input trajectories u_{k+i}^C can also be approximated by a discrete time polynomial function.

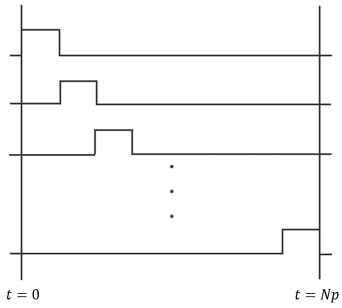


Figure 6. The control input of the MPC in discrete time can be considered as a combination of pulse functions

5.2. Laguerre Basis Functions

5.2.1. Constructions

The discrete time Laguerre functions are derived from the discretization of continuous-time Laguerre functions [20]. In this paper, the state space form of Laguerre functions will be implemented. Let the initial Laguerre function be:

$$L_0 = \sqrt{1-a^2}\begin{bmatrix}1 & -a & a^2 & -a^3 & \dots & (-1)^{N-1}a^{N-1}\end{bmatrix}^{\mathrm{T}}, \tag{14}$$

where a is the pole of the discrete time Laguerre function and $0 \leq a < 1$. N shows the division of Laguerre vectors. Let, $L_k = \begin{bmatrix}l_k^1 & l_k^2 & \dots & l_k^N\end{bmatrix}^{\mathrm{T}}$, then,

$$L_{k+1} = A_l L_k, \tag{15}$$

where the matrix $A_l(N \times N)$ is a function of a. For example, for $N = 3$,

$$A_l = \begin{bmatrix} a & 0 & 0 \\ \sqrt{1-a^2} & a & 0 \\ -a\sqrt{1-a^2} & \sqrt{1-a^2} & a \end{bmatrix}, \quad L_0 = \sqrt{1-a^2}\begin{bmatrix} 1 \\ -a \\ a^2 \end{bmatrix} \tag{16}$$

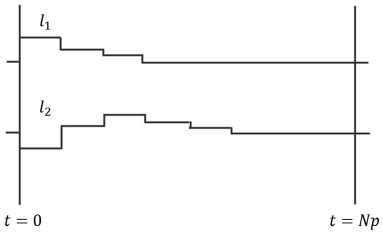


Figure 7. An example of Laguerre functions in discrete time with $a = 0.5$

Laguerre functions are orthonormal functions, therefore it has the following prop-

erty,

$$\sum_{k=0}^{N_p} l_k^i l_k^j = 0 \quad for \quad i \neq j,$$

$$\sum_{k=0}^{N_p} l_k^i l_k^j = 1 \quad for \quad i = j.$$

5.2.2. Control Inputs

At the beginning of this section, we mentioned that the control inputs in MPC are represented by a set of pulse functions. Here, the Laguerre functions are used to describe the control inputs:

$$u_k^C = \sum_{i=1}^{N} c^i l_k^i. \tag{17}$$

In a vector form,

$$U_k^C = L_k^{\mathrm{T}} C_l, \tag{18}$$

where

$$C_l = \begin{bmatrix} c^1 & c^2 & ... & c^N \end{bmatrix}^{\mathrm{T}}$$

and c are parameters which will be derived by solving the cost function.

If Laguerre functions are used to approximate the control inputs, the new control inputs in Eq. (18) are C_l. Now the length of the control inputs is the number of Laguerre functions used (N) while in the conventional MPC, the length of the control inputs of Eq. (12) is N_c. As a single Laguerre function carries more information in comparison with the pulse function, fewer number of Laguerre functions are required to formulate the input, $N < N_c$. As a result, the length of the control input is shortened and the computational load is also decreased.

5.2.3. Motion with Laguerre Functions

The MPC with Laguerre basis functions is closely related to the conventional MPC. If we put Eq. (18) in Eq. (2) and Eq. (4), the future states of the CoM and feet can be generated in the form of Laguerre functions as:

$$\hat{X}_{k+N_p}^C = A_c^{N_p} \hat{X}_k^C + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1} B_c L_i^T C_l, \tag{19}$$

$$Y_{k+N_p}^C = C_c A_c^{N_p} \hat{X}_k^C + \sum_{i=0}^{N_p-1} C_c A_c^{N_p-i-1} B_c L_i^T C_l. \tag{20}$$

5.2.4. Selection of Parameters

There are two explicit tuning parameters in the design of Laguerre functions which are the pole of the discrete time Laguerre function a, and the number of Laguerre functions, N. The parameter a determines the configuration of Laguerre functions. If the control horizon N_c is known, a can be selected in the area near $a \approx exp(-\frac{k_l}{N_c})$ where k_l is set between 5 to 10 according to the number of Laguerre functions that are applied. However, for the MPC used in the WPG, it is recommended to select a small value of a, due to the steep change of the control input that appears in the double support phase. In order to guarantee feasible solutions, a small value of a is our first choice.

The number of terms N indicates how many Laguerre functions are applied to approximate the target. The complexity of the calculation is influenced by the choice of N. As N increases, a better description of the target is provided while calculational load is also added. A small value of N means a fast calculation however it may lead to an unfeasible solution. At the beginning, N can be chosen with the same value as N_c and then designers can reduce N until an acceptable balance between accuracy and efficiency is achieved.

5.2.5. Notes

(1) Fast and steep changes (happened in double support phase) in the control inputs can be handled with Laguerre polynomials.
(2) It is easy to tune the performance of the MPC with Laguerre functions as there are two explicit parameters which are a and N.
(3) The MPC with Laguerre functions is the same as the traditional MPC when $a = 0$ and $N = N_c$.
(4) The complexity of solving the cost function is reduced due to the reduction in the length of control inputs.

5.3. Haar Basis Functions

Besides applying Laguerre basis functions, Haar basis functions can also be used to represent the control input.

5.3.1. Construction

Let the initial Haar function h_0 be the signal which assumes a constant value on the unit interval [0,1) as follows:

$$h_0 = \frac{1}{(\sqrt{2})^m} \tag{21}$$

where m is an integer. From the point of view of signal representation, a large m indicates smaller resolution and higher frequency.

A set of Haar wavelets are orthogonal functions defined as,

$$h_{m,k} = \frac{1}{(\sqrt{2})^m} \psi \left(\frac{1}{2^m} t - k \right) \tag{22}$$

where k relates to the phase shift of the wavelets and $\psi(t)$ is the indicator function

$$\psi(t) = \begin{cases} 1 & \text{if } t \in [0, \frac{1}{2}) \\ -1 & \text{if } t \in [\frac{1}{2}, 1) \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

Haar wavelets are orthogonal and square-integrable in ℓ_2 norm. The approximation of the control input can be achieved by applying Haar series.

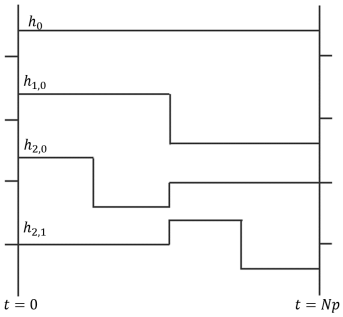


Figure 8. Haar basis functions.

5.3.2. Transformation from Pulse basis

The input signal is represented by a finite discrete form of Eq. (12), and it can be expressed with Haar functions by

$$U_k^C = \sum_{i=1}^{2^l-1} \delta_{0,k}\phi_{0,k} = \sum_{k=0}^{2^{l-m-1}} \sum_{m=1}^{l-1} c_{m,k}h_{m,k} \tag{24}$$

where δ, c are coefficients for pulse basis and Haar basis respectively, and ϕ is the shifting pulse. Changing the input signal from the standard pulse function into the Haar basis is a unitary transformation. The transformation matrix H is orthogonal, $H^T H = I$,

$$U_k^C = H^T C_h \tag{25}$$

$$C_h = H U_k^C \tag{26}$$

where U_k^C, C_h are the matrices of coefficients δ, c respectively. For example, if $m = 2$,

$$\begin{bmatrix} c_{0,0} \\ c_{1,0} \\ c_{2,0} \\ c_{2,1} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \delta_{0,0} \\ \delta_{1,0} \\ \delta_{2,0} \\ \delta_{3,0} \end{bmatrix} \tag{27}$$

Figure 8 shows the Haar basis functions for $h_0, h_{1,0}, \{h_{2,k}\}_{k=0,1}$.

5.3.3. Blocking

Besides giving acceptable performances, a control algorithm should be computationally efficient for inexpensive on-line implementation. Blocking is a strategy to reduce the size of optimization problems. In this method, the number of vectors to be calculated is decreased by projecting the vector space onto a lower subspace. As a result, the insignificant linear combinations of the input are eliminated in the calculation. Let the input vector after the blocking be U_B and block matrix B_{blk},

$$U_B = B_{blk}U \tag{28}$$

B_{blk} is an orthogonal matrix and it is designed to eliminate some of the rows that are related to the particular linear combinations of the input which are set to zero a priori.

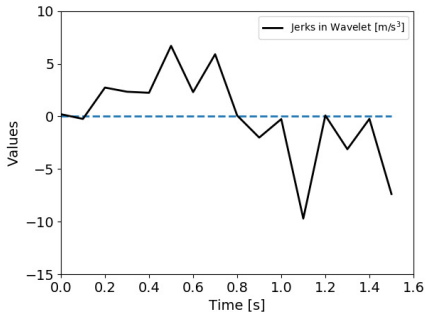


Figure 9. Jerks after transformation from time domain (Figure 5) to wavelet domain, where it is easier to find zeros in the input that can be blocked.

Though blocking is an efficient way to reduce computational load, it is hard to apply blocking directly in MPC; since the proper choice of blocking matrix is usually obscure in time domain. Here, we propose a general method to use the blocking matrix, that is representing the control input in wavelet domain with Haar functions. In Figure 9, wavelet transformation provides a new perspective to view the input, where the blocking techniques can be implemented in an insightful and convenient way. As the transformation from time domain to wavelet domain is a unity one, we do not loss any information from the input. Therefore the accuracy is preserved.

Here is an example of blocking. Suppose the control input in wavelet domain is U_w which has a dimension of 8,

$$U_w = [u_{0,0}, u_{1,0}, u_{2,0}, u_{2,1}, u_{3,0}, u_{3,1}, u_{3,2}, u_{3,3}]^T \tag{29}$$

and blocking matrix B_{blk} is designed as

$$B_{blk} = \begin{bmatrix} |1| & 0 & 0 & 0 \\ 0 & |1| & 0 & 0 \\ 0 & 0 & |1| & 0 \\ 0 & 0 & |0| & 0 \\ 0 & 0 & 0 & |1| \\ 0 & 0 & 0 & |0| \\ 0 & 0 & 0 & |0| \\ 0 & 0 & 0 & |0| \end{bmatrix} \tag{30}$$

The variables after blocking, U_B is calculated by Eq. (28),

$$U_B = [u_{0,0}, u_{1,0}, u_{2,0}, u_{3,0}]^T \tag{31}$$

As a result, the length of the input vector is shortened from 8 to 4.

5.3.4. Motion with Haar Functions

If we put Eq. (25) in Eq. (2) and Eq. (4), the future states of the CoM and feet are built in the form of Haar functions as

$$\hat{X}^C_{k+N_p} = A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1} B_c H_i^T C_h, \tag{32}$$

$$Y^C_{k+N_p} = C_c A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} C_c A_c^{N_p-i-1} B_c H_i^T C_h, \tag{33}$$

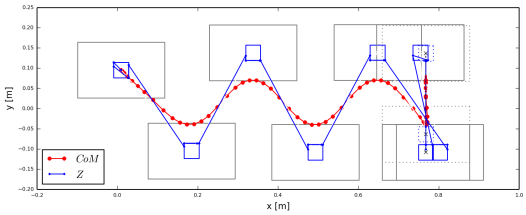
Blocking matrix can be added to eliminate part of the control input.

$$\hat{X}^C_{k+N_p} = A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1} B_c B_{blk} H_i^T C_h, \tag{34}$$

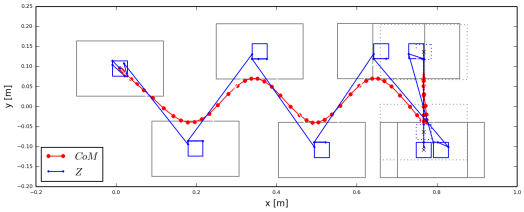
$$Y^C_{k+N_p} = C_c A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} C_c A_c^{N_p-i-1} B_c B_{blk} H_i^T C_h, \tag{35}$$

5.3.5. Notes

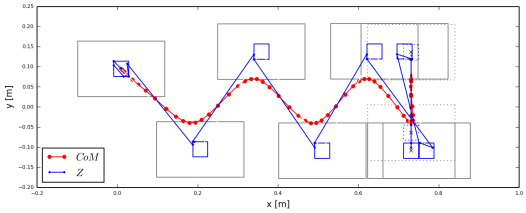
(1) The transformation of signals from time domain to wavelet domain is a unity one. Thus information is kept when transforming the input from time to wavelet domain.
(2) After transforming an input into wavelet domain, a blocking matrix can be introduced to reduce the dimension of the input.
(3) The MPC with Haar functions is the same as the conventional MPC when blocking is not implemented.



(a) Patterns by conventional MPC



(b) Patterns by MPC with Haar functions



(c) Patterns by MPC with Laguerre functions

Figure 10. Forward walking patterns. MPC with Haar functions generates the most accurate trajectory while MPC with Laguerre functions runs faster.

Table 1. Walking forward with approximation of jerks of x.

Controller	Dimension of input	Errors	Time(ms)	Time saving
MPC	16	-4.7%	21.9	0
MPC with Haar	13	-0.2%	18.7	-14.6%
MPC with Laguerre	10	-7.5%	15.4	-29.7%

6. Simulation

In this section, the walking gaits produced by the conventional MPC [15], MPC with Laguerre and Haar basis functions are compared. The walking pattern generator is tested on the HRP-2 humanoid robot. A step is made regularly every 0.8s where 0.7s for single support and 0.1s time duration for double support. The time of prediction for all the MPC tested is set to 1.6s which means it predicts two full steps in the future. Three walking strategies are tested which are walking forward, sidewalking and walking with feet rotations. The humanoid robot starts walking with the right foot, then tracks the targeting velocity and finally take 2s to reach a rest position, which is a double support condition.

The robot walks forward with a velocity of $Vel_{k+1}^{ref} = \begin{bmatrix} 0.2 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$ for 4s. The three components in Vel_{k+1}^{ref} represent the CoM's velocity along X, Y direction and angular velocity along vertical axis respectively. At the beginning, the jerks of the CoM along x axis $U_k^{C_x}$ are represented by basis functions. In Figure 10, all the MPC based WPGs generated feasible trajectories.

From Table 1, it can be seen that the running time of the three controllers is different. In the conventional MPC, computational cost is 21.9ms while the MPC with Haar and the MPC with Laguerre save 14.6% and 29.7% of calculation time respectively, in comparison with conventional MPC. The numbers of the control inputs $U_k^{C_x}$ in the MPC, MPC with Haar and MPC with Laguerre are $16, 13, 10$ respectively. The reduction in the numbers of the control inputs leads to a faster calculation. The tuning parameters of the MPC with Laguerre are, $a = 0.3, N = 10$.

Although the MPC with Haar does not work as fast as the MPC with Laguerre, it provides the most accurate result among the three tested MPCs while the computa-

Table 2. Walking forward with modification of jerks of y.

Controller	Dimension of input	Errors	Time(ms)	Time saving
MPC	16	-4.7%	21.9	0
MPC with Haar	12	-0.2%	22.3	+1.8%

Table 3. Sidewalking.

Controller	Dimension of input	Errors	Time(ms)	Time saving
MPC	16	-3.8%	18.2	0
MPC with Haar	5	-3.3%	10.2	-43.9%
MPC with Laguerre	1	-3.3%	8.2	-54.9%

tional cost is less compared with the conventional MPC.

In addition to operations on the jerk of x, $U_k^{C_x}$, we then apply basis functions to represent the jerk of y, $U_k^{C_y}$. However, in the case of implementing basis functions in $U_k^{C_y}$, the MPC of Laguerre can not generate a feasible trajectory. The MPC with Haar can produce stable foot trajectory (see Figure 11), but the running time of it, is higher than that of conventional MPC, see Table 2.

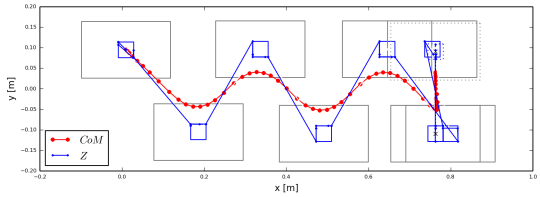


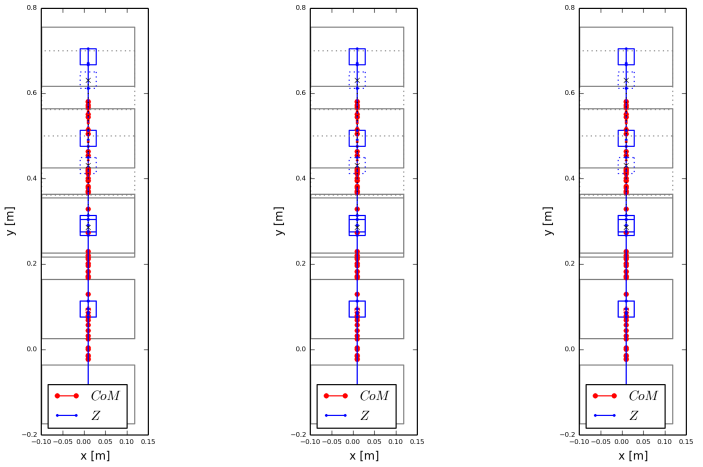
Figure 11. Walking patterns generated by approximation of both jerks of x and y with Haar basis functions.

The reason for such performances is that there are both a peak and a low in the jerk of y (see Figure 5), which makes it difficult for basis functions to approximate. In order to design a fast and stable realtime WPG, only $U_k^{C_x}$ is represented with basis functions.

In the case of sidewalking, see Figure 12, the MPCs with basis functions have a huge improvement compared with traditional MPC. Table 3 shows that the reference velocity is assigned to 0.2m/s and the running time of process is reduced by 43.9% in the MPC with Haar and by 54.9% in the MPC with Laguerre functions. More than half of the calculation time is saved when the Laguerre functions are used. The reason of improvement in both MPC with basis functions is that when the robot walks in y direction, the CoM's trajectory along x axis does not change much. Therefore, the control inputs along x direction can be greatly reduced and it results in a fast calculation.

To demonstrate the behavior of the orientation controller, a constant velocity $V_{k+1}^{ref} = [0.2, 0, 0.2]$ including rotation, is sent to the walking pattern generator for 4s. Both the position and the orientation controller are designed with basis functions. The setting for the position controller with basis functions is the same as in the case of walking forward. The control inputs, which are jerks along left and right direction, in the orientation controller are approximated by basis functions.

Table 4 shows the length of the input and the calculation time of three orientation controllers. Simulation shows that the orientation controller with Laguerre functions has a quick response and it reduces by 46.2% the calculation time. By applying Haar



(a) MPC (b) MPC with Haar functions (c) MPC with Laguerre functions

Figure 12. Sidewalking patterns of the three MPCs tested. From left to right, MPC, MPC with Haar functions and MPC with Laguerre functions. Both of the MPC with basis functions work faster than the conventional MPC.

Table 4. Orientation Controller.

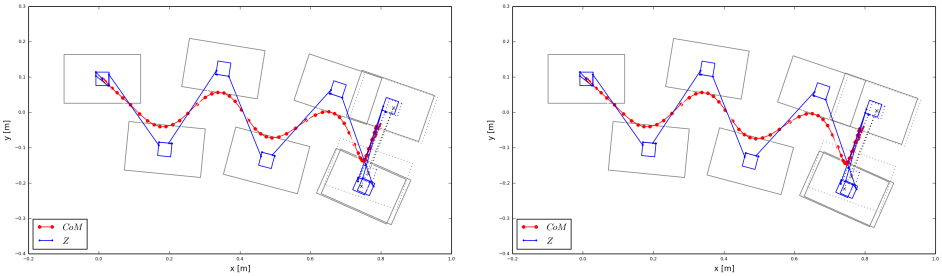
Controller	Dimension of input	Orientation errors	Time(ms)	Time saving
MPC	16	-4.8%	10.8	0
MPC with Haar	14	-5.7%	8.8	-18.5%
MPC with Laguerre	9	-7.3%	5.8	-46.2%

basis functions, the calculation time is saved by 18.5% compared with conventional MPC. In Figure 13, all the tested MPCs generated feasible walking patterns while the MPC with basis functions work faster.

In order to test the efficiency of our proposed method, simulations and experiments are done. The simulation is shown in Figure 14 where the HRP-2 robot walked forward with a velocity of 0.2m/s. Experiments using the HRP-2 robot with a reference velocity of $V_{k+1}^{ref} = [0.2, 0, 0]$ and $V_{k+1}^{ref} = [0.2, 0, -0.2]$ are shown in Figure 15 and Figure 16, respectively.

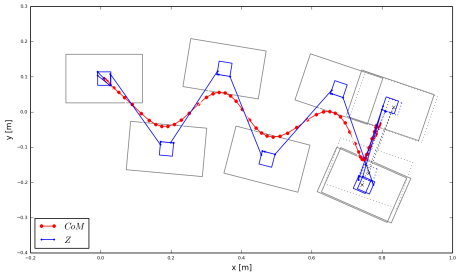
7. Conclusion

The main contribution of this paper is introducing basis functions into the design of control inputs for online WPG based on MPC. The control input represented by basis functions, like Haar functions and Laguerre functions, requires less free variables therefore the computational efficiency is improved. Simulations and experiments showed that feasible CoM trajectories can be generated while computational time can be reduced in both orientation and position controller when using basis functions. The MPC with Laguerre basis functions has the fastest performance because the configuration of Laguerre functions can be tuned to fit the property of the control input. We also showed that the MPC with Haar functions can always provide feasible and accurate solutions. The basis functions are not restricted to Laguerre functions and Haar functions, researchers can choose different basis functions for various purposes.



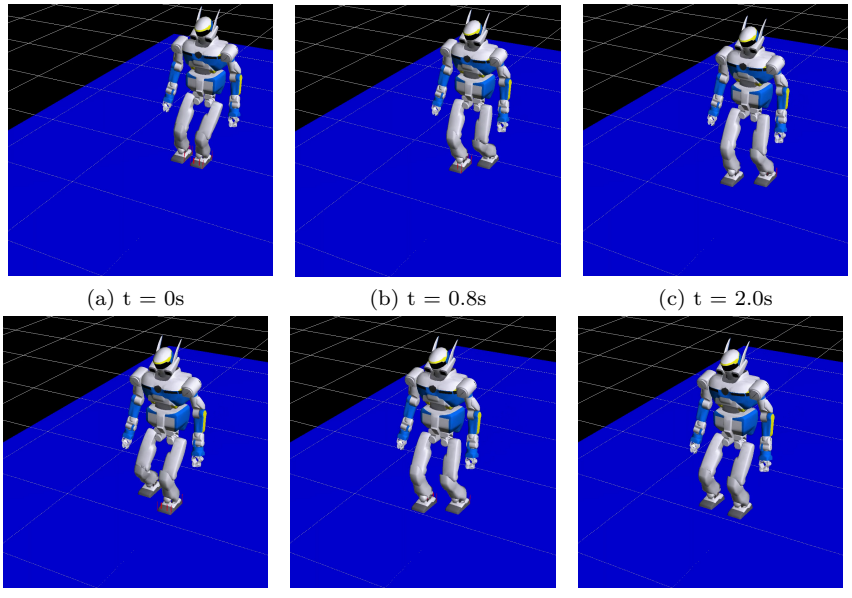
(a) Patterns by MPC

(b) Patterns by MPC with Haar functions



(c) Patterns by MPC with Laguerre functions

Figure 13. Foot trajectories when rotation is involved. MPC with Laguerre functions is the fastest controller.



(a) t = 0s (b) t = 0.8s (c) t = 2.0s

(d) t = 3.0s (e) t = 4.0s (f) t = 5.0s

Figure 14. Bipedal walking using the WPG with Laguerre basis functions.

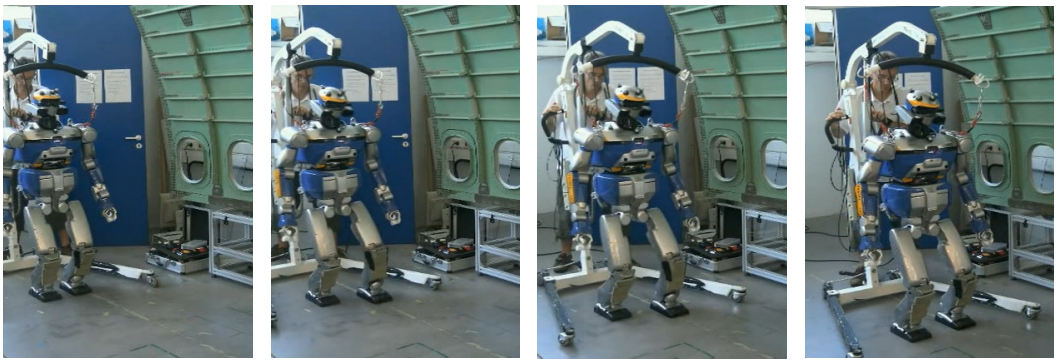


Figure 15. The experiment of walking forward when using the WPG with Laguerre basis functions.

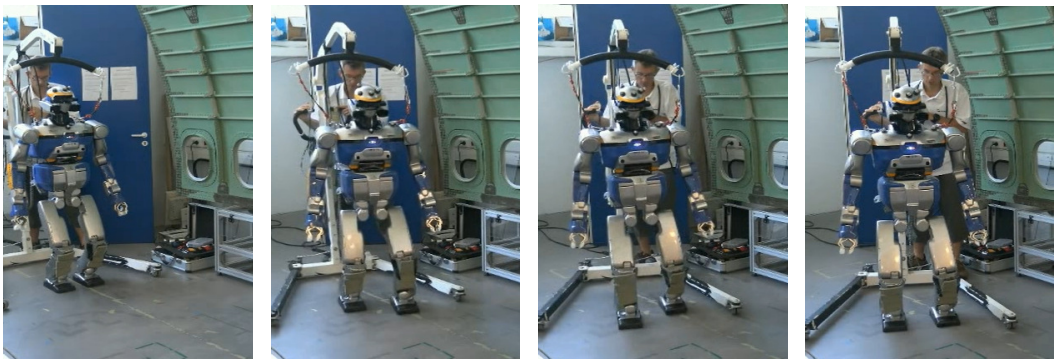


Figure 16. The experiment of walking forward and turning right when using the WPG with Laguerre basis functions.

8. Future Work

We would like to implement more basis functions to approximate control input. More detailed analysis about experiment results will also be done. With a fast walking pattern generator, we can let humanoids challenge more complex tasks while walking, like manipulation and co-operations with humans.

References

[1] Grey, M. X., Garrett, C. R., Liu, C. K., Ames, A. D. and Thomaz, A. L.: Humanoid manipulation planning using backward-forward search, *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, pp. 5467–5473 (2016).

[2] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Yokoi, K. and Hirukawa, H.: Biped walking pattern generation by a simple three-dimensional inverted pendulum model, *Advanced Robotics*, Vol. 17, No. 2, pp. 131–147 (2003).

[3] Wieber, P.-B.: On the stability of walking systems, *Proceedings of the international workshop on humanoid and human friendly robotics* (2002).

[4] Bohórquez, N. and Wieber, P.-B.: Adaptive step duration in biped walking: a robust approach to nonlinear constraints, *IEEE RAS International Conference on Humanoid Robots 2017* (2017).

[5] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K. and Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point, *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, Vol. 2, IEEE, pp. 1620–1626 (2003).

[6] Harada, K., Kajita, S., Kaneko, K. and Hirukawa, H.: An analytical method for real-time gait planning for humanoid robots, *International Journal of Humanoid Robotics*, Vol. 3, No. 01, pp. 1–19 (2006).

[7] Morisawa, M., Harada, K., Kajita, S., Kaneko, K., Sola, J., Yoshida, E., Mansard, N., Yokoi, K. and Laumond, J.-P.: Reactive stepping to prevent falling for humanoids, *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, IEEE, pp. 528–534 (2009).

[8] Nishiwaki, K. and Kagami, S.: Trajectory design and control of edge-landing walking of a humanoid for higher adaptability to rough terrain, *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, pp. 3432–3439 (2012).

[9] Takasugi, N., Kojima, K., Nozawa, S., Okada, K. and Inaba, M.: 3D walking and skating motion generation using divergent component of motion and gauss pseudospectral method, *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, IEEE, pp. 5003–5010 (2017).

[10] Englsberger, J., Ott, C., Roa, M. A., Albu-Schäffer, A. and Hirzinger, G.: Bipedal walking control based on capture point dynamics, *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, IEEE, pp. 4420–4427 (2011).

[11] Englsberger, J., Ott, C. and Albu-Schäffer, A.: Three-dimensional bipedal walking control based on divergent component of motion, *IEEE Transactions on Robotics*, Vol. 31, No. 2, pp. 355–368 (2015).

[12] Krause, M., Englsberger, J., Wieber, P.-B. and Ott, C.: Stabilization of the capture point dynamics for bipedal walking based on model predictive control, *IFAC Proceedings Volumes*, Vol. 45, No. 22, pp. 165–171 (2012).

[13] Romualdi, G., Dafarra, S., Hu, Y. and Pucci, D.: A Benchmarking of DCM Based Architectures for Position and Velocity Controlled Walking of Humanoid Robots, *arXiv preprint arXiv:1809.02167* (2018).

[14] Herdt, A., Diedam, H., Wieber, P.-B., Dimitrov, D., Mombaur, K. and Diehl, M.: Online walking motion generation with automatic footstep placement, *Advanced Robotics*, Vol. 24,

No. 5-6, pp. 719–737 (2010).
[15] Herdt, A., Perrin, N. and Wieber, P.-B.: Walking without thinking about it, *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, IEEE, pp. 190–195 (2010).
[16] Naveau, M., Kudruss, M., Stasse, O., Kirches, C., Mombaur, K. and Souères, P.: A reactive walking pattern generator based on nonlinear model predictive control, *IEEE Robotics and Automation Letters*, Vol. 2, No. 1, pp. 10–17 (2017).
[17] Shafiee-Ashtiani, M., Yousefi-Koma, A. and Shariat-Panahi, M.: Robust bipedal locomotion control based on model predictive control and divergent component of motion, *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, pp. 3505–3510 (2017).
[18] Van Donkelaar, E. T., Bosgra, O. H. and Van den Hof, P. M.: Model predictive control with generalized input parametrization, *Control Conference (ECC), 1999 European*, IEEE, pp. 1693–1698 (1999).
[19] Muehlebach, M. and D'Andrea, R.: Parametrized infinite-horizon model predictive control for linear time-invariant systems with input and state constraints, *American Control Conference (ACC), 2016*, IEEE, pp. 2669–2674 (2016).
[20] Wang, L.: *Model predictive control system design and implementation using MATLAB®*, Springer Science & Business Media (2009).
[21] Misra, S., Reddy, R. and Saha, P.: Model predictive control of resonant systems using Kautz model, *International Journal of Automation and Computing*, Vol. 13, No. 5, pp. 501–515 (2016).
[22] Wang, R., Tippett, M. J. and Bao, J.: Fast Wavelet-Based Model Predictive Control of Differentially Flat Systems, *Processes*, Vol. 3, No. 1, pp. 161–177 (2015).
[23] Chen, M., Zhang, A. and Chong, K. T.: A Novel Controller Design for Three-phase Voltage Source Inverter, *International Journal of Control, Automation and Systems*, Vol. 16, No. 5, pp. 2136–2145 (2018).
[24] Lee, J. H., Chikkula, Y., Yu, Z. and Kantor, J. C.: Improving computational efficiency of model predictive control algorithm using wavelet transformation, *International Journal of Control*, Vol. 61, No. 4, pp. 859–883 (1995).
[25] Wahlberg, B. and Mäkilä, P.: On approximation of stable linear dynamical systems using Laguerre and Kautz functions, *Automatica*, Vol. 32, No. 5, pp. 693–708 (1996).
[26] Orin, D. E., Goswami, A. and Lee, S.-H.: Centroidal dynamics of a humanoid robot, *Autonomous Robots*, Vol. 35, No. 2-3, pp. 161–176 (2013).