Robotica (2025), **43**, pp. 1394–1410
doi:10.1017/S0263574725000347

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Task priority-based redundancy resolution of a 20 DoFs humanoid robot for object pick and place with minimum energy and dynamic balance

James Sorokhaibam and Ashish Dutta

Department of Mechanical Engineering, Indian Institute of Technology Kanpur, Kanpur, Uttar Pradesh, India
Corresponding author: James Sorokhaibam; Email: james@ittk.ac.in

Received: 16 August 2024; **Revised:** 17 December 2024; **Accepted:** 9 February 2025; **First published online:** 28 March 2025

Keywords: humanoid robots; redundant manipulators; robot dynamics; motion planning

Abstract

Humanoid robots are highly redundant, and finding whole-body optimal trajectories for various tasks is very complex. This paper proposes a method to find an energy-optimal, dynamically balanced, and collision-free trajectory of the 20 degrees of freedom humanoid robot in pick and place application. The task of pick and place is divided into three subtasks using the Pseudoinverse Jacobian method of redundancy resolution. The three subtasks are end effector trajectory represented by $\mathcal{T}_1$, hip trajectory represented by $\mathcal{T}_2$, and maximizing the manipulability represented by $\mathcal{T}_3$. The Pseudoinverse Jacobian method is coupled with particle swarm optimization (PSO) to find the optimal trajectories. The main contribution of this paper is the decomposition of the whole-body task of the humanoid robot into three distinct subtasks to find energy-optimal, dynamically balanced, and obstacle-free trajectories. The concept of virtual surface is used to avoid dragging objects on the table surface. The problem is optimized with Particle Swarm Optimization. Simulations were conducted to pick up and place objects from a table and constrained spaces like a drawer. The results show that the robot can pick and place objects from defined locations on the table.

1. Introduction

Humanoid robots possess high dexterity, making them valuable in various applications such as navigating cluttered environments, handling objects, and even folding clothes. Research efforts have extensively focused on refining the capabilities of these robots, especially in tasks such as object manipulation in cluttered environments. Kuffner et al. [1] introduced a sampling-based method that utilizes Rapidly exploring random trees (RRTs) to plan dynamically stable and collision-free trajectories for reaching an object by a humanoid robot. By employing "AutoBalancer," a dynamics filter function, this approach efficiently generates a final trajectory that ensures both dynamic stability and collision avoidance. Dalibard et al. [2] explored the use of RRTs to create collision-free paths, which were then optimized to achieve dynamic stability and natural, human-like motion in humanoids during for variety of applications like reaching an object. Teja et al. [3] divided the humanoid into two sub-chains, utilizing RRT* to generate a statically stable and collision-free optimal path for one sub-chain while computing joint positions for the other using inverse kinematics. An extension to the bilateral Fast Marching Trees in the form of waypoints is used in Balachandran et al. [4] for whole-body motion planning of humanoid robots. The goal configuration in the above method is determined by gradient-based inverse kinematics coupled with the optimization process. The upper body motion of a humanoid robot is generated in Ruchanurucks [5] using B-splines based functions and limits over various physical attributes like joint limit, torque limit, and collisions are set by appropriate coefficients of the B-splines. A smooth interpolation method of the variable-time interval is proposed in Yu et al. [6] to generate the collision-free and energy-jerk-minimal trajectory by using quintic B-spline curve and normal distribution.

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

Utilizing the redundancy present in redundant robots through the application of the Jacobian's null space projection facilitates task decomposition, enabling the allocation of distinct subtasks to the robots. The concept of task priority was introduced in Nakamura et al. [7], where a required task is divided into subtasks according to the order of priority. By employing pseudoinverses of Jacobian matrices, the joint rates of the robot manipulator are determined, enabling the execution of lower-priority subtasks using redundancy without compromising the higher-priority subtasks. Ott et al. [8] harnessed redundancy for dynamic control in the seven degree of freedoms (DoFs) compliant arm of the humanoid robot Justin, employing null-space projection techniques to handle multiple prioritized tasks. Gams et al. [9] utilize task decomposition for real-time full body motion imitation on COMAN robot where the primary task is set as maintaining the dynamic balance based on ZMP criteria, and the secondary task is set as motion imitation. Karimi and Ahmadi [10] focus on implementing strict task priority allocation for kinematic control and obstacle avoidance on a five DoFs planar manipulator and on a seven DoFs biped robot. An adaptive weighted Newton iteration is used in Liu et al. [11] for energy efficient trajectory considering the kinematic constraints on a 6 DoFs manipulator. The task decomposition method is used in Tomic et al. [12] for generating human-like motion on robot ROMEO by imitating the motion in Cartesian and joint spaces, obtained from the imitation algorithm. Quan et el. [13] utilized the idea of redundancy angle discretization on a seven DoFs modular manipulator and proposed the concept of the self-motion space of the trajectory to obtain collision-free solutions for a fixed Cartesian trajectory. An optimal collision-free trajectory for redundant manipulators considering higher mechanical advantage was achieved in Zacharia and Xidias [14] by setting an optimal sequence of the subtasks: cycle time, collision avoidance, and the manipulability index using Genetic Algorithm coupled with the Bump-Surface concept.

Zhao and Wei [15] used the gradient projection of the hierarchical planning strategy into the null-space of Jacobian to generate human-like motion of the joint trajectory in seven DoFs Robai robotic arm in reaching and grasping of object. A framework for controlling the acceleration in Cartesian space to manage the execution of operational tasks and constraints for human-like motion on Honda humanoid robot model was introduced in Hammam et al. [16] by incorporating a dynamically consistent redundancy resolution method to minimize joint motions. A multi-priority control approach for asymmetric coordination in redundant dual arm was introduced in Zhang et al. [17] by dividing the control into the main task and the secondary task. The main task control involves regulating the errors of the two end effectors and their relative errors by constructing a model of spatial parallel spring and damping, which establishes the dual-arm coordination impedance relationship in Cartesian space. The secondary task control focuses on optimizing the performance of redundant joint impedance and avoiding joint limits in the null space. Jeong and Chang [18] apply the Lagrangian multiplier method to address the task-priority challenge in a 28 DoFs humanoid robot, using the square error of the end-effector acceleration as the criterion function. The subtasks are arranged in decreasing priority order, encompassing joint limit, static balance, left foot position, and the motion of a fencing sword striking a target. This approach enables the precise execution of the fencing sword subtask.

Tsuichihara et al. [19] solve inverse kinematics of humanoid robot HRP-4 for the task of reaching low-height positions by both hands in real time. The orientation of the torso was calculated using a regressor and then obtained final position of the center of mass (CoM) for a given target configuration. In each control step, the author first computes the joint angles of the lower body from the CoM position, feet position, and torso orientation, and then the joint angles of both arms. Sentis and Khatib [20] use redundancy resolution for task prioritization to establish a hierarchy between a set of tasks by giving the highest priority to constraint-handling tasks such as balance control, joint limit, and collision avoidance, while projecting operational tasks such as hand control and foot control in the null space and then controlling the posture with the available redundancy. Task priority is used in Acar and Murakami [21] for the control task of dynamic balance and end effector of a two-wheeled mobile manipulator. The problem of balance of humanoid robots during object handling while standing is tackled in Tsuichihara et al. [22] using the weighted pseudo-inverse of Jacobian. The work in Sugiura et al. [23] addresses the self-collision problem in real time by applying a virtual force in the direction of collision and also by

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

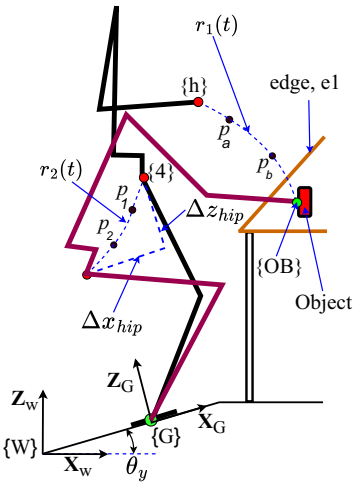


Figure 1. *Hip and end-effector trajectory.*

using a dynamic task prioritizing method that effectively combines prioritization for reaching a target and avoiding collisions.

The existing work in humanoid robots fails to address the energy expenditure of humanoid robots during pick and place. This paper focused on finding the minimum number of subtasks and determining energy optimal trajectory by considering the dynamics of the whole body robot without compromising the dynamic balance and end-effector manipulability during pick and place application from table to floor. The motion of humanoid robots is highly complex due to the branched chain and redundancy present. To reduce this complexity and also to solve the energy-optimal problem simpler, the Jacobian method of task decomposition/redundancy resolution is used in our work. Setting too many subtasks increases the complexity of the problem and conflict between the subtasks arises. Setting a lesser number of subtasks might reduce the complexity, but it will not be able to fulfill our main goal. Our work deals with the three distinct task decomposition of whole-body humanoid robots to find energy-optimal, dynamically balanced, and obstacle-free trajectories for pick-and-place applications. The whole task of pick and place is divided into three stages stage I (approach), stage II (pick), and stage III (place). Each stage is solved using the method of Pseudoinverse of Jacobian matrix for redundancy resolution on task priority basis where the various constants are optimized using PSO. The end effector of the robot, i.e., the task hand tracking a desired trajectory, $r_1(t)$ is considered as primary task $\mathcal{T}_1$, and the hip of the robot, i.e., origin of frame {4} tracking a desired trajectory $r_2(t)$ is chosen as secondary task $\mathcal{T}_2$ as shown in Figure 1 and maximizing the manipulability of the robot as tertiary task $\mathcal{T}_3$. Task $\mathcal{T}_1$ is of the highest priority, followed by task $\mathcal{T}_2$ and task $\mathcal{T}_3$ has the least priority. Newton–Euler (**NE**) for tree type robot is used to solve the dynamics of the humanoid robot. The main contribution of this paper is finding an energy-optimal, obstacle-free, and dynamically balanced whole-body trajectory and also maximizing the manipulability $V(q)$ Kang et al. [24] of the end effector as the robot moves along the desired trajectory. The optimal number of subtasks to be set during pick and place application is discussed in Sections 5.2, 5.3, and 5.4. The structure of the paper is as outlined below: Section 2 details the methodology for generating desired trajectories, Section 3 discusses the dynamics of tree-type robotic system, and Section 4 presents the generation of trajectories that are energy optimal, obstacle-free, and dynamically balanced using PSO. The simulation and results are presented in Section 5, while Section 6 concludes the paper.

2. Methodology

The kinematic model of the full-body humanoid is shown in Figure 2a. The whole body is divided into three sub-chains: primary chain, secondary chain, and tertiary chain. The right leg, torso, and right hand

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

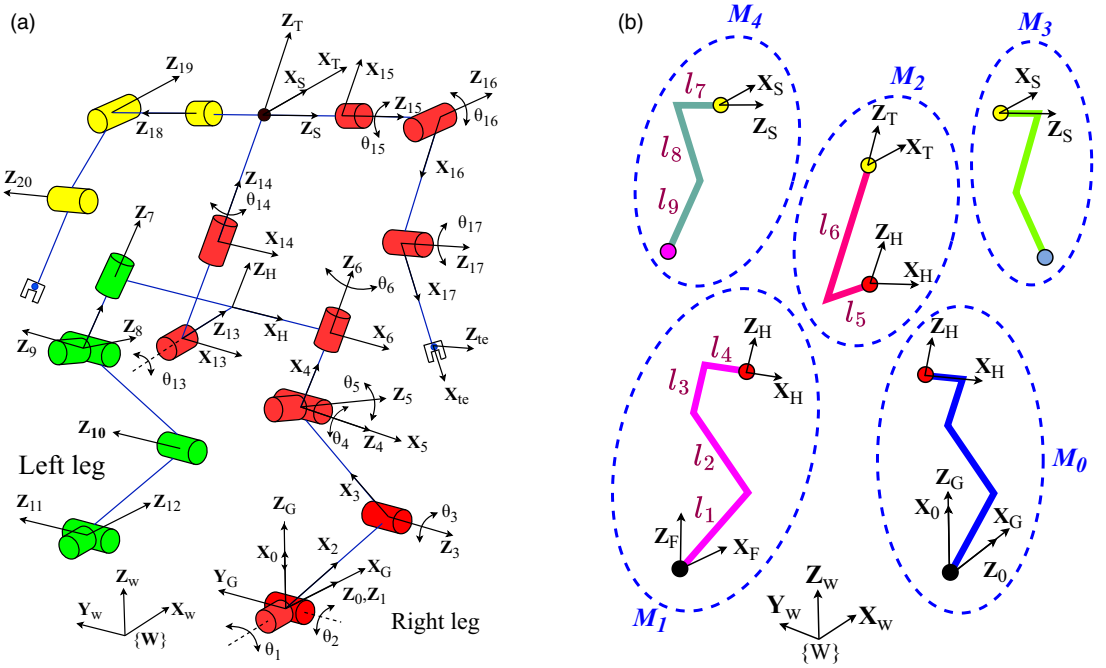


Figure 2. *(a) 20 degrees of freedom kinematic model. (b) Modules architecture of the full-body humanoid.*

form the primary chain, the left leg forms the secondary chain, and the left-hand forms the tertiary chain. The joints associated with the frames {1}, {5}, and {6} of the primary chain and that of the frame {7}, {8}, and {12} of the secondary chain are locked to make the hip motion planer. So, the primary chain has only eight active joints. The joint rates of the primary chain are computed using the pseudoinverse method Eq. (3) and that of the secondary chain is calculated using the geometrical method of inverse kinematics at each time step. All the joints of the tertiary chain are also locked while performing the task. The tree-type robotic system is also divided into 5 different modules as shown in Figure 2b. Frame $\{G\}$ is attached to the base of module 0, M_0, and Frame $\{F\}$ is attached to the end of module 1, M_1.

The location and orientation of the humanoid robot are defined by

- The location and orientation of the ground frame $\{G\}$ with respect to the world frame $\{W\}$ denoted by ${}^{W}P_G \in \mathbb{R}^3$ and ${}^{W}\boldsymbol{\Theta}_G \in \mathbb{R}^3$, respectively.
- The location and orientation of the frame $\{F\}$ with respect to the frame $\{G\}$ denoted by ${}^{G}P_F \in \mathbb{R}^3$ and ${}^{G}\boldsymbol{\Theta}_F \in \mathbb{R}^3$ respectively.

$r_1(t) = \begin{bmatrix} x_1(t) & y_1(t) & z_1(t) \end{bmatrix}^T$ and $r_2(t) = \begin{bmatrix} x_2(t) & y_2(t) & z_2(t) \end{bmatrix}^T$ are vector-valued functions where the components are composite splines. Three cubic splines are connected at two via points to form the composite spline by ensuring velocity and acceleration continuity. $p_a \in \mathbb{R}^3$, $p_b \in \mathbb{R}^3$ are the two via points for $r_1(t)$ and $p_1 \in \mathbb{R}^3$, $p_2 \in \mathbb{R}^3$ for $r_2(t)$ as shown in Figure 1. Δx_{hip} and Δz_{hip} define the hip displacement while performing the whole task . Out of the eight DoFs of the primary chain, task $\mathcal{T}_1$ uses a minimum of three DoFs, task $\mathcal{T}_2$ uses two DoFs, and the remaining DoFs are used for the task of maximizing the manipulability index $V(q)$ of the end effector. The algorithm works in an iterative way where the input required for finding the joint rates using the method of Pseudoinverse of the Jacobian matrix is obtained by the best output values from the PSO algorithm. The following assumptions are made:

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

- Initially, the robot is in a statically balanced state.
- Body of the robot is aligned with the unit vector $\hat{Z}_W$ initially.
- The location of the robot is such that the end effector is able to reach the object at the extreme end of the table without compromising the dynamic balance.
- The object is massless, and there is no force of interaction when the robot picks the object.

The task $\mathcal{T}_1$ assigned to the end effector (task hand) of the robot is such that the end effector must reach the target (a fixed point) depending on the stage (stage I, stage II, or stage III). As the task $\mathcal{T}_2$ assigned to the robot is primarily responsible for maintaining the balance of the robot, the values Δx_{hip} and Δz_{hip} are such that the zero moment point (ZMP) Vukobratovic and Borovac [25] of the whole body humanoid always lies inside the foot support polygon. The above analogy and assumptions clearly shows that task $\mathcal{T}_1$ will always be fulfilled and $\mathcal{T}_1$ will be of higher priority than $\mathcal{T}_2$. The remaining degree of freedom will be used for maximizing the manipulability end effector if possible as the robot fulfills the task $\mathcal{T}_1$ and $\mathcal{T}_2$.

The initial and end conditions required for computing $r_1(t)$ and $r_2(t)$ are mentioned in Eq. (14), Eq. (15) and Eq. (12), Eq. (13), respectively. The values of $p_a, p_b, p_1, p_2, \Delta x_{hip}$, and Δz_{hip} are best value output from the PSO algorithm. Joint rates computed using Eq. (3) are such that the robot fulfills the task $\mathcal{T}_1$, $\mathcal{T}_2$, and $\mathcal{T}_3$. The joint rates obtained from Eq. (3) are used to compute the energy consumed in doing the task by solving the dynamics of the whole body robot. Energy-optimal, dynamically balanced, and obstacle-free trajectories are obtained by optimizing using PSO. The fitness function is a function of energy consumed with the constraint of collision penalty and ZMP violation penalty in the optimization algorithm.

The end-effector trajectory $\dot{r}_1$ and hip trajectory $\dot{r}_2$ relate to the joint rate vector $\dot{q} \in \mathbb{R}^8$ by Eq. (1) and Eq. (2)

$$\dot{r}_1 = J_1 \dot{q} \tag{1}$$
$$\dot{r}_2 = J_2 \dot{q} \tag{2}$$

where,

$$\dot{q} = \begin{bmatrix} \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_{13} & \dot{\theta}_{14} & \dot{\theta}_{15} & \dot{\theta}_{16} & \dot{\theta}_{17} \end{bmatrix}^T$$
$$\dot{r}_1 = \begin{bmatrix} \dot{x}_1(t) & \dot{y}_1(t) & \dot{z}_1(t) \end{bmatrix}^T$$
$$\dot{r}_2 = \begin{bmatrix} \dot{x}_2(t) & \dot{y}_2(t) & \dot{z}_2(t) \end{bmatrix}^T.$$

Here, Jacobian $J_1 \in \mathbb{R}^{3\times 8}$ and Jacobian $J_2 \in \mathbb{R}^{3\times 8}$. J_2 is written as $J_2 = [J_{leg}|\mathbf{0}]$, where $J_{leg} \in \mathbb{R}^{3\times 2}$ relates joint rates $\dot{\theta}_2$ and $\dot{\theta}_3$ with hip velocity. General solution for $\dot{q}$ of Eq. (1) and Eq. (2) that performs primary, secondary, and tertiary tasks is given by Eq. (3), Yoshikawa [26]

$$\dot{q}_d = J_1^{\dagger}\dot{r}_1 + \tilde{J}_2^{\dagger}(\dot{r}_2 - J_2 J_1^{\dagger}\dot{r}_1) + (I_8 - J_1^{\dagger}J_1 - \tilde{J}_2^{\dagger}\tilde{J}_2)k \tag{3}$$

where, $\dot{q}_d$ is the vector of the desired joint rates. The Jacobian matrices are computed as follows Witney [27]

$$J = \begin{bmatrix} J^1 & J^2 & \dots & J^n \end{bmatrix} \tag{4}$$

where,

$$J^i = \left(\hat{z}_{i-1} \times p_n^{i-1}\right). \tag{5}$$

$J^i \in \mathbb{R}^3$ in Eq. (5) is the i^{th} column vector of the Jacobian matrix Eq. (4), $\hat{z}_{i-1}$ is a unit vector in the direction of z axis of $(i-1)^{th}$ frame and $p_n^{i-1} \in \mathbb{R}^3$ is a vector from the origin of $(i-1)^{th}$ frame to that i^{th} frame. The augmented Jacobian $\tilde{J}_2$ in Eq. (3) is given by $\tilde{J}_2 = J_2(I_8 - J_1^{\dagger}J_1)$, where $J^{\dagger} = J^T(JJ^T)^{-1}$ is the Moore–Penrose pseudo-inverse of J. The vector k in Eq. (3) is given by

$$k = K_p \boldsymbol{\xi} \tag{6}$$

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

where $\boldsymbol{\xi} \in \mathbb{R}^8$ is the manipulability gradient vector given by Eq. (8), $K_p \in \mathbb{R}^{8\times 8}$ is the appropriate diagonal weight matrix Eq. (7), and the diagonal entries are obtained through optimization.

$$K_p = \begin{bmatrix} k_1 & 0 & \dots & 0 \\ 0 & k_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & k_8 \end{bmatrix} \tag{7}$$

$$\boldsymbol{\xi} = \begin{bmatrix} \xi_1 & \xi_2 & \dots & \xi_8 \end{bmatrix}^T \tag{8}$$

where

$$\xi_l = \frac{\partial}{\partial q_l} V(q), V(q) = \sqrt{det(J_1 J_1^T)}, l = 1, 2, 3, \dots, 8, \tag{9}$$

$$\xi_l = \frac{V(q)}{2} tr\left((JJ^T)^{-1} \left(\frac{\partial J}{\partial q_l} J^T + J \frac{\partial J^T}{\partial q_l} \right) \right) \tag{10}$$

$$\xi_l = \frac{1}{2} V(q) \sum_{i,j=1}^{3} a_{ij} (J_{1il} J_{1j}^T + J_{1jl} J_{1i}^T) \tag{11}$$

$V(q)$ is the manipulability of the end effector of the primary chain of the robot, a_{ij} is the $(i,j)^{th}$ element of $(J_1 J_1^T)^{-1}$, J_{1i} is the i^{th} row vector of J_1, and J_{1il} is the partial differential of J_{1i} with respect to q_l.

The initial and end conditions for the hip trajectory are given by Eq. (12) and Eq. (13). As there is no hip motion in the frontal plane during the task, the value of Δy_{hip} is zero. Figure 1 shows the hip trajectory and the via points. Initially, the hip is at $r_{hip_i} = \begin{bmatrix} x_{hip_i} & y_{hip_i} & z_{hip_i} \end{bmatrix}^T$, where r_{hip_i} is the initial hip position at the beginning of a stage and t_c is the cycle time for each stage.

$$r_2(0) = r_{hip_i} \tag{12}$$

$$r_2(t_c) = r_{hip_i} + \begin{bmatrix} \Delta x_{hip} & 0 & \Delta z_{hip} \end{bmatrix}^T \tag{13}$$

where

$$r_{hip_i} = {}^W R_0^T ({}^W P_4 - {}^W P_G)$$
$$\Delta x_{hip} = \Delta x_{hip} \hat{X}_G$$
$$\Delta z_{hip} = \Delta z_{hip} \hat{Z}_G$$

The initial and end conditions for the end effector trajectory $r_1(t)$ are given by Eq. (14) and Eq. (15)

$$r_1(0) = {}^W R_0^T ({}^W P_h - {}^W P_G) \tag{14}$$

$$r_1(t_c) = {}^W R_0^T ({}^W P_{OB} - {}^W P_G) \tag{15}$$

where the rotation matrix ${}^W R_0$ rotates the frame {W} to frame {0}, and ${}^W P_G$ is the position vectors of the frames {G} with respect to frame {W}. ${}^W P_4$ and ${}^W P_h$ are the position vectors of the hip and the end effector at the beginning of a stage, and ${}^W P_{OB}$ is the target location all with respect to the frame {W} see Figure 1.

3. Dynamics of tree-type robotic system

A full-body humanoid robot is a tree-type robotic system comprising various modules where each module is a serial chain as shown in Figure 2b. Module M_0 is the base module of the system, and frame {**0**} attached to the module M_0 is the base of the whole robot. Velocities and accelerations are computed

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

Algorithm 1 Kinematics for tree type system.

1: **Module 0 Kinematics**
2: **INPUT** $\boldsymbol{\omega}_0 = \mathbf{0}, \dot{\boldsymbol{\omega}}_0 = \mathbf{0}, \dot{v}_0 = -g, nf = 7$
3: **OUTPUT** $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \dot{v}, \dot{v}_c, F^0, N^0$
4: $\boldsymbol{\omega}_H = \boldsymbol{\omega}_{nf}, \dot{\boldsymbol{\omega}}_H = \dot{\boldsymbol{\omega}}_{nf}, \dot{v}_H = \dot{v}_{nf}$
5: **Module 1 Kinematics**
6: **INPUT** $\boldsymbol{\omega}_0 = \boldsymbol{\omega}_H, \dot{\boldsymbol{\omega}}_0 = \dot{\boldsymbol{\omega}}_H, \dot{v}_0 = \dot{v}_H, nf = 6$
7: **OUTPUT** $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \dot{v}, \dot{v}_c, F^1, N^1$
8: **Module 2 Kinematics**
9: **INPUT** $\boldsymbol{\omega}_0 = \boldsymbol{\omega}_H, \dot{\boldsymbol{\omega}}_0 = \dot{\boldsymbol{\omega}}_H, \dot{v}_0 = \dot{v}_H, nf = 3$
10: **OUTPUT** $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \dot{v}, \dot{v}_c, F^2, N^2$
11: $\boldsymbol{\omega}_T = \boldsymbol{\omega}_{nf}, \dot{\boldsymbol{\omega}}_T = \dot{\boldsymbol{\omega}}_{nf}, \dot{v}_T = \dot{v}_{nf}$
12: **Module 3 Kinematics**
13: **INPUT** $\boldsymbol{\omega}_0 = {}^S\!R_T\boldsymbol{\omega}_T, \dot{\boldsymbol{\omega}}_0 = {}^S\!R_T\dot{\boldsymbol{\omega}}_T$
$\dot{v}_0 = {}^S\!R_T\dot{v}_T, nf = 3$
14: **OUTPUT** $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \dot{v}, \dot{v}_c, F^3, N^3$
15: **Module 4 Kinematics**
16: **INPUT** $\boldsymbol{\omega}_0 = {}^S\!R_T\boldsymbol{\omega}_T, \dot{\boldsymbol{\omega}}_0 = {}^S\!R_T\dot{\boldsymbol{\omega}}_T$
$\dot{v}_0 = {}^S\!R_T\dot{v}_T, nf = 3$
17: **OUTPUT** $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \dot{v}, \dot{v}_c, F^4, N^4$

using iterative outward algorithm Craig [28] from the innermost link to the outermost link of each module. For the inter-modular velocity and acceleration propagation, velocities and accelerations available at the end of each module serve as input to the next adjoining module. Algorithm 1 shows the kinematics of a tree-type robotic system. ${}^S\!R_T$ rotates the frame $\{T\}$ with respect to frame $\{S\}$. Here, $\boldsymbol{\omega}_j$, $\dot{\boldsymbol{\omega}}_j$, and $\dot{v}_j$ are the angular velocity, angular acceleration, and linear acceleration of the origin of frame $\{j\}$, respectively. $\dot{v}_{c_j}$ is the acceleration of the center of mass of j^{th} link. nf is the number of frames in a module excluding the initial base frame of the module required for solving the kinematics and dynamics. F^m and N^m are read as inertial force vector and inertial torque vector of all the links of module M_m where $m = 0 \ldots 4$.

Forces and Torques are computed using recursive **Newton-Euler** algorithm (**NE**) [28]. The inter-modular force and torque propagation for the tree type is shown in Algorithm 2. ${}^H\!R_{13}$ and ${}^H\!R_7$ are the rotation matrices that rotates the frame {13} and frame {7} with respect to frame $\{H\}$ respectively. ${}^H\!P_{13}$ and ${}^H\!P_7$ are the position vectors of the frame {13} and frame {7} with respect to frame $\{H\}$, respectively. f_j^m and $\boldsymbol{\tau}_j^m$ are the net force and net torque on the j^{th} link of module M_m respectively. Work done (E) by the robot for a stage is given by Eq. (16), where t_c is the total time taken for a stage. $\boldsymbol{\tau}_i(t)$ and $\dot{\boldsymbol{\theta}}_i$ are the torque and joint rate for the i^{th} link, respectively. The ground reaction force denoted by f_{ground} present in line number 14 of the algorithm is taken as half of the weight of the robot.

$$E = \sum_{i=1}^{20} \int_0^{t_c} \tau_i(t)\dot{\theta}_i(t)dt \tag{16}$$

ZMP of the whole robot on inclined plane Sardain and Bessonnet [29] is computed by the relation shown in Eq. (17). n denotes the unit vector normal to the surface on which the robot stands. M^{g_i} and F^{g_i} are resultant moment and resultant force respectively, due to the combined effect of gravity and inertia forces.

$$P_{ZMP} = \frac{n \times M^{g_i}}{F^{g_i} \cdot n} \tag{17}$$

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

Algorithm 2 Dynamics for tree type system.

Module 4 Dynamics
  INPUT $f_{in} = \mathbf{0}, \boldsymbol{\tau}_{in} = \mathbf{0}, nf = 4$
  Use **NE** to compute $f^4, \boldsymbol{\tau}^4$
  $f_S^4 = f_1^4, \boldsymbol{\tau}_S^4 = \boldsymbol{\tau}_1^4$
Module 3 Dynamics
  INPUT $f_{in} = \mathbf{0}, \boldsymbol{\tau}_{in} = \mathbf{0}, nf = 4$
  Use **NE** to compute $f^3, \boldsymbol{\tau}^3$
  $f_S^3 = f_1^3, \boldsymbol{\tau}_S^3 = \boldsymbol{\tau}_1^3$
Module 2 Dynamics
  INPUT $f_{in} = {}^T R_S(f_S^4 + f_S^3)$,
    $\boldsymbol{\tau}_{in} = {}^T R_S(\boldsymbol{\tau}_S^4 + \boldsymbol{\tau}_S^3), nf = 2$
  Use **NE** to compute $f^2, \boldsymbol{\tau}^2$
  $f_H^2 = {}^H R_{13} f_1^2, \boldsymbol{\tau}_H^2 = {}^H R_{13} \boldsymbol{\tau}_1^2 + {}^H P_{13} \times {}^H R_{13} f_1^2$
Module 1 Dynamics
  INPUT $f_{in} = f_{ground}, \boldsymbol{\tau}_{in} = \mathbf{0}, nf = 6$
  Use **NE** to compute $f^1, \boldsymbol{\tau}^1$
  $f_H^1 = {}^H R_7 f_1^1, \boldsymbol{\tau}_H^1 = {}^H R_7 \boldsymbol{\tau}_1^1 + {}^H P_7 \times {}^H R_7 f_1^1$
Module 0 Dynamics
  INPUT $f_{in} = f_H^1 + f_H^2$,
    $\boldsymbol{\tau}_{in} = \boldsymbol{\tau}_H^1 + \boldsymbol{\tau}_H^2, nf = 6$
  Use **NE** to compute $f^0, \boldsymbol{\tau}^0$

4. Hand and hip trajectory computation using PSO

For every end-effector trajectory $r_1(t)$, hip trajectory $r_2(t)$ and matrix K_p, joint rates are computed in such a way the manipulability of the end effector of the robot is maximized. Energy-optimal obstacle-free trajectories are obtained by optimizing using PSO. The input to the optimization algorithm is given by Eq. (18), where k_{opt} vector is the diagonal entries of the matrix K_p in Eq. (7). Particle Swarm Optimization (PSO) toolbox available in MATLAB is used here for the optimization. The syntax is given by $x = particleswarm(fun, nvars, lb, ub)$, where (fun) is the function to be optimized, (nvars) is the number of variables, (lb) is the lower bound, and (ub) is the upper bound. Here, p_a and p_b contain three independent variable each. The motion of the hip is confined only in the sagittal plane so p_1 and p_2 has two independent variables each. Number of independent variables in k_{opt} is discussed in Sections 5.1. Fitness function Eq. (19) required for the optimization algorithm is the sum of work done (E) given by Eq. (16) and penalty for constraints violation, β given by Eq. (20).

$$X_{opt} = \left[p_a \; p_b \; p_1 \; p_2 \; \Delta x_{hip} \; \Delta z_{hip} \; k_{opt}\right]^T \tag{18}$$

$$F = E + \beta \tag{19}$$

$$\beta = \sum_{i=1}^{pn} \beta_i \tag{20}$$

Penalty, β is imposed during optimization for violating the constraints and pn is the number of constraints imposed. Various constraints used here are

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

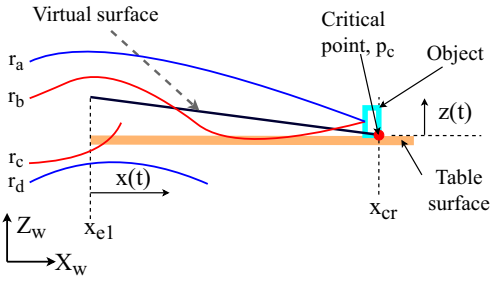


Figure 3. *Virtual surface.*

- ZMP constraints and
- collision constraints between the object and robot links.

RMS value of the deviation of ZMP of the whole-body humanoid around the point P_{sz} located inside the support polygon multiplied by a scalar γ_1 is taken as a penalty. ZMP penalty calculated for the whole instance is shown in Eq. (21) where *vp* is the number of time instances.

$$\beta_1 = \gamma_1 \frac{1}{2} \sqrt{\sum_{i=1}^{vp} \| P_{zmp_i} - P_{sz} \|^2} \tag{21}$$

Various penalties are imposed when the robot trajectories interfere with the obstacles. The penalties imposed for obstacle avoidance are divided as penalties for end effector, elbow, shoulder, and neck collision with the edge of the table represented as β_2, β_3, β_4, and β_5, respectively, and penalties for forearm collision with the edge of the table as β_6. Penalties for end effector, elbow, shoulder, and neck collision with the surface of the table represented as β_7, β_8, β_9, and β_{10}, respectively, and that of task hand collision with the body of the robot as β_{11}. Penalties for collision with the edge, e_1 of the table Figure 1 become effective when the shortest distance d_i of the robot part from the edge of the table is less than the threshold distance, d_{min} where γ_2 is a scalar Eq. (22). d_{min} is introduced in order to avoid the value of β_2 becoming very small when d_2 approaches zero. d_{min} is taken as 10 cm in our work.

$$\beta_2 = \begin{cases} \gamma_2(d_{min} + d_2), & \text{if} \quad d_2 < d_{min} \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

The end-effector trajectory, elbow trajectory, shoulder trajectory, and neck trajectory must not interfere with the surface of the table. For any trajectory $r(t) = f(x(t), y(t), z(t))$, let z_{cross} be the set of $z(t)$ such that the respective $x(t)$ lie in the region (x_{edge}, z_{cross}) i.e. $z_{cross} = \{z(t) | x(t) \in (x_{e1}, x_{edge})\}$. For such a trajectory to be feasible, any element from the set z_{cross} must exclusively lie above the virtual surface created above the table or below the surface of the table. In Figure 3, r_a and r_d are feasible trajectories, whereas r_b and r_c are not feasible trajectories. The virtual surface in Figure 3 given by Eq. (23) is created to avoid sliding of the end effector on the table surface or dragging of the object on the table surface. Penalty β_6 can be checked by measuring the shortest distance between the forearm and edge, e_1 of the table Figure 1. Task hand collision with the body of the robot is checked by imposing a constraint on the joint position associated with the frame {16} and penalty β_{11} is imposed when the constraint is violated. The complete schematic for finding the energy optimal, obstacle-free, and dynamically balanced full body trajectory that fulfills the primary, secondary, and tertiary task is shown in Figure 4. The algorithm has two parts, PSO part and fitness value generation. It works iteratively until optimal trajectories are achieved. It starts with the initial output from the PSO algorithm where the output will be used to generate the fitness value. The vector X_{opt} contains the variable of the optimization of swarm size 100, and a tolerance value of 10^{-3} is used for the optimization. The output from the PSO part is used to generate the desired trajectories for the swarm that fulfill all three subtasks. Work done and ZMP are

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

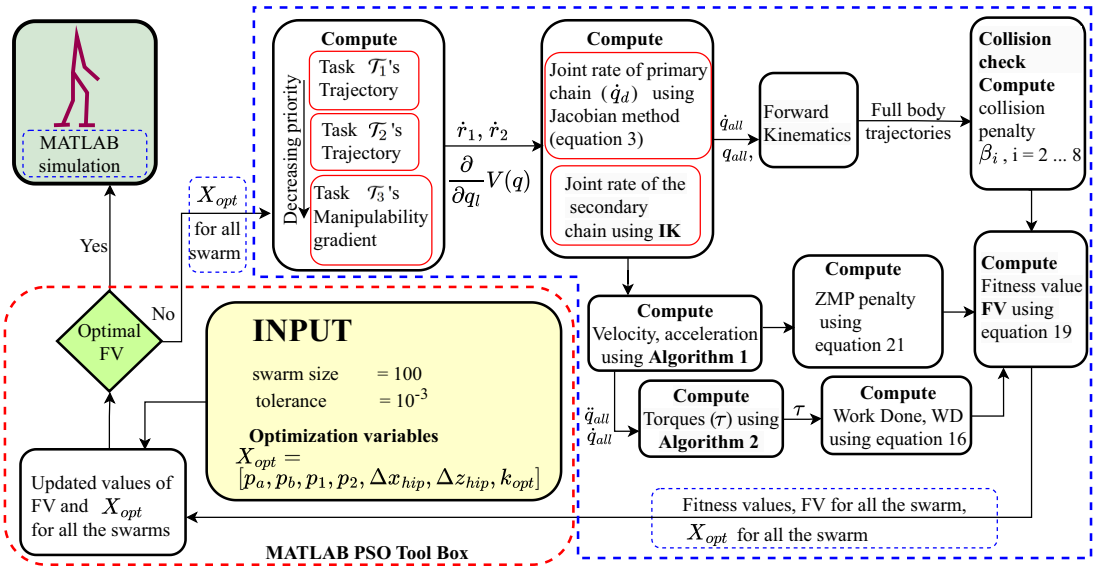


Figure 4. *Schematic for generating the energy-optimal, collision-free, and dynamically balanced trajectory for the full-body humanoid robot that fulfills all the primary, secondary, and tertiary tasks.*

further computed using the dynamics of the tree-type robotic system. Fitness values are computed using Eq. (19). The vector X_{opt} along with the generated fitness values for all the swarms are further fed to the PSO part for improving the solution and the iteration goes on until energy optimal, collision-free, and dynamic balanced trajectories are obtained.

$$z(x) = \left(\frac{x_{cr} - x(t)}{x_{cr} - x_{e1}} \right) \tag{23}$$

5. Simulation and results

The simulation of the robot was conducted in MATLAB R2022b on a 12^{th} Gen Intel(R) Core(TM) i7-12700 2.10 GHz 16.0 GB RAM Window 11 pro machine. The cycle time t_c for each stage is 2 s. The optimization was done using the PSO toolbox available in MATLAB and was run for a swarm size of 100 and function tolerance of 10^{-3}. The length and mass of the different parts of the robot are shown in Table I. Each foot has a mass of 1 kg, length 30 cm, and breadth 12 cm, and the total mass of the robot is 54 kg.

5.1. The value of k_{opt}

For three subtasks decomposition case as in Section 5.2, after assigning the subtask $\mathcal{T}_1$ and $\mathcal{T}_2$, three DoFs still remains in the null-space. These extra DoFs are used for maximizing the manipulability $V(q)$. Finding an appropriate value of the vector k_{opt} as mentioned in Section 4 is the key to determining an energy optimal body configuration. The components of the vector k_{opt} is modified as $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 \end{bmatrix}^T$. The intuition behind this is, gradient of $V(q)$ for $q = \theta_2$ and $q = \theta_3$ do not contribute in the manipulability maximization. So, the vector k in Eq. (6) is written as $k = \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 k_3 & \xi_4 k_4 & \xi_5 k_5 & \xi_6 k_6 & \xi_7 k_7 & \xi_8 k_8 \end{bmatrix}^T$.

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

Table I. *Length and mass of parts of the robot. Figure 2b shows the meaning of each symbols.*

Parts	Length (cm)	Mass (kg)
$l_1 \times 2$	45	2
$l_2 \times 2$	45	2
$l_3 \times 2$	10	2
$l_4 \times 2$	12.5	2.5
$l_5 \times 1$	5	1
$l_6 \times 1$	40	24
$l_7 \times 2$	22.5	1
$l_8 \times 2$	30	2
$l_9 \times 2$	30	2

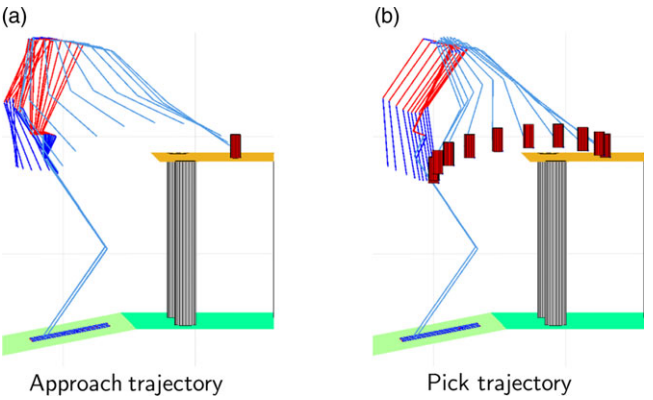


Figure 5. *Trajectories for* $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 \end{bmatrix}$.

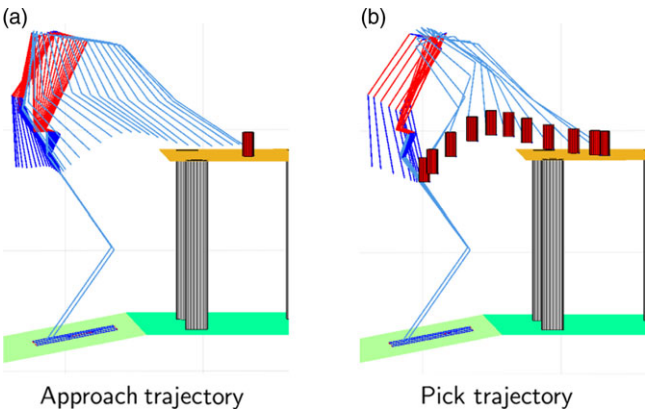


Figure 6. *Trajectories for* $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 \end{bmatrix}$.

The optimization algorithm for $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 \end{bmatrix}$ was run for several iterations and results in same pattern in the trajectories. Figure 5a and 5b shows the trajectories for approach and pick, respectively. The same is repeated for Figure 6. Undesirable motion patterns which is a case of elbow up position are seen in Figures 5b and 6b. These undesirable patterns are avoided by setting the weights k_6, k_7, and k_8 equal. So k_{opt} is modified as $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_6 & k_6 \end{bmatrix}$. So, k_{opt} contains only four independent variables number of optimization variables, *nvars* mentioned in Section 4 is 16. Trajectories for approach, pick, and place, and the respective optimization curves are shown in Figure 7a, Figure 7b,

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

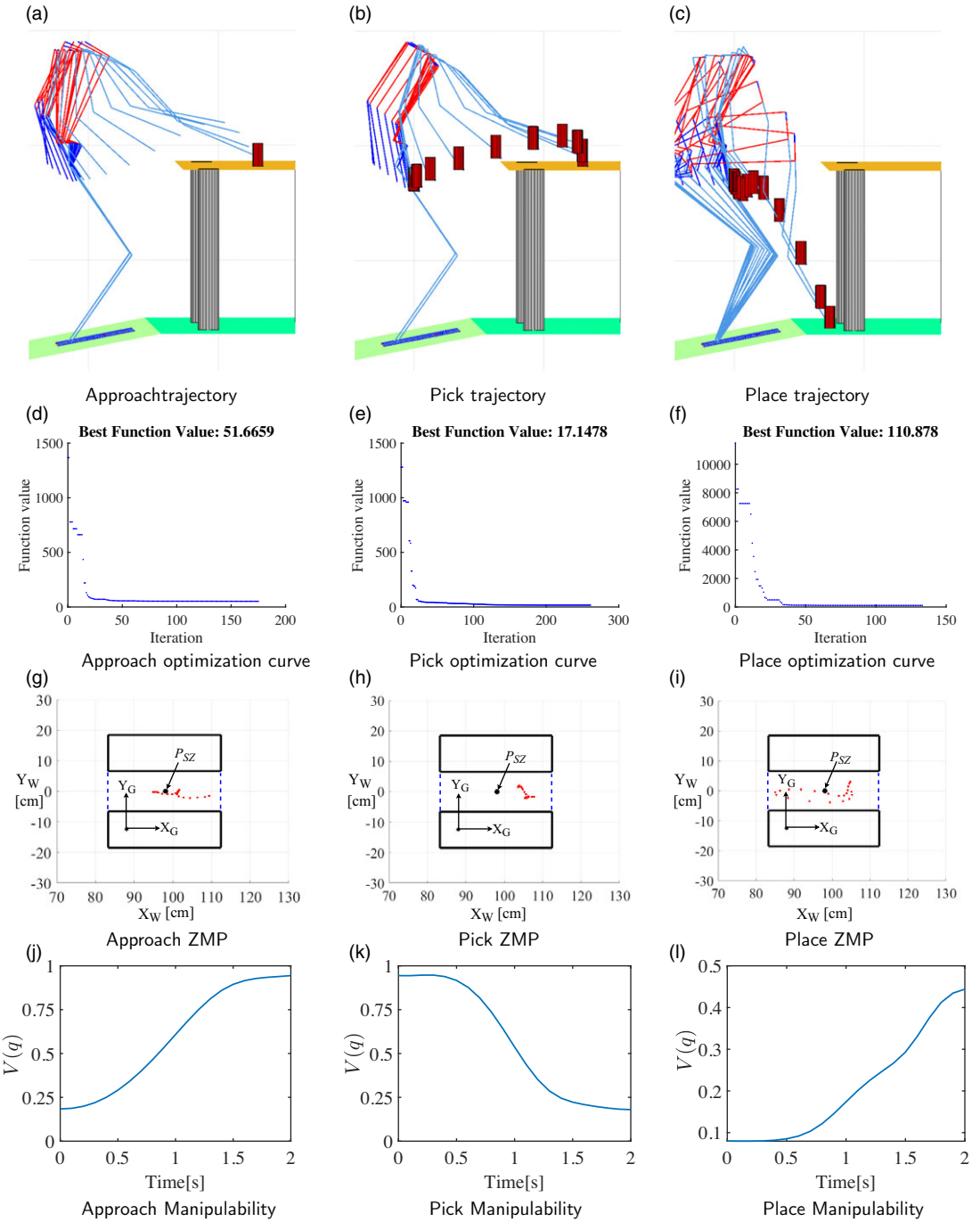


Figure 7. *Pick and place for* $k_{opt} = \begin{bmatrix} 1 & 1 & k_3 & k_4 & k_5 & k_6 & k_6 & k_6 \end{bmatrix}$.

and Figure 7c. The objective function Eq. (19) is a function of work done and penalties for violating collision constraints and ZMP constraints. Initially, when the optimization begins, penalty terms dominate over the work done term. Work done term starts dominating when the curve begins to converge and finally only the work done term remains when the curve is fully converged. Figure 7d, 7e, and 7f shows optimization curves during approach, pick, and place, respectively, with respective work done as

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

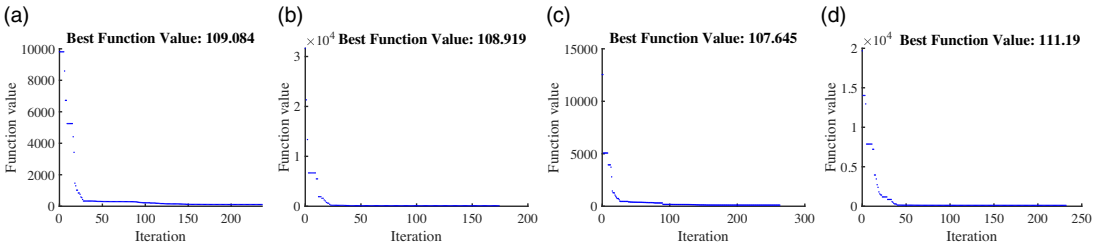


Figure 8. *Different optimization curve during place (stage III) for* $k_{opt} = [1\ 1\ k_3\ k_4\ k_5\ k_6\ k_6\ k_6]$.

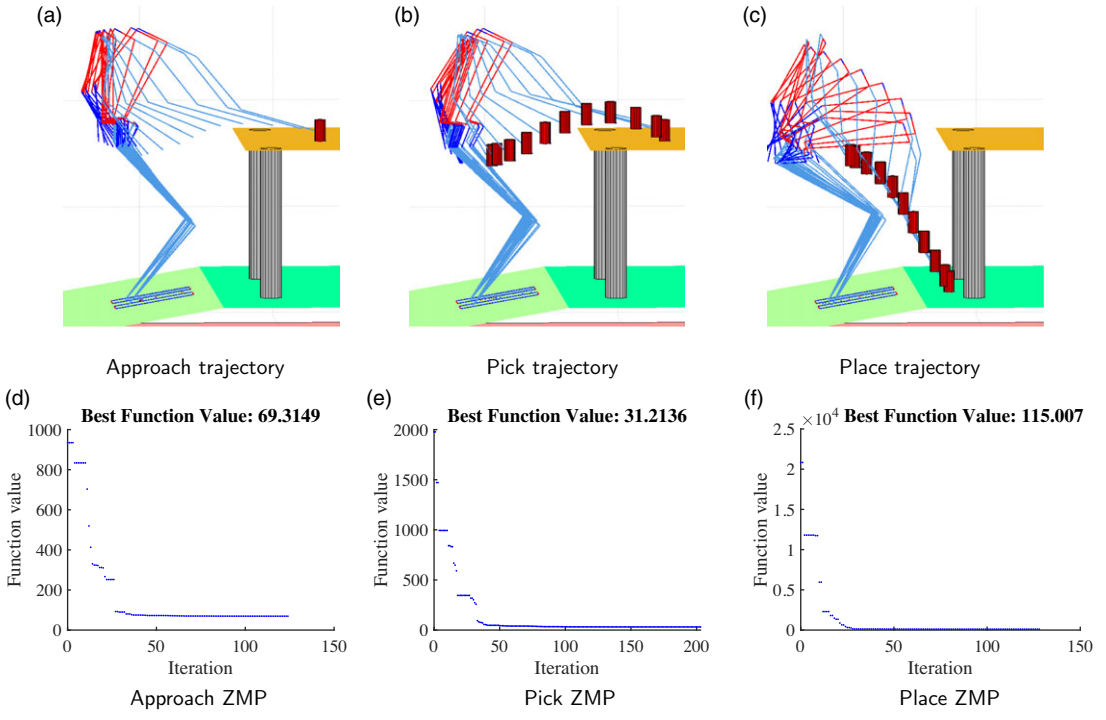


Figure 9. *Two task ($\mathcal{T}_1$ and $\mathcal{T}_3$) decomposition trajectory and ZMP.*

51.67 joules, 17.15 joules, and 110.9 joules. In Eq. (21), the value of $\gamma_1 = 0$ when the ZMP is inside the support polygon otherwise $\gamma_1 = 10$.

The optimization algorithm for placing the object on the floor for the same object's location was run several times and approximately results in the same values as shown in Figure 8.

5.2. Case I: three task decomposition $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$.

The trajectories for the three different stages (approach, pick, and place) of the task are shown in Figure 7a, 7b, and 7c, respectively. In all three stages, the robot was able to perform the primary task of reaching the target without hitting the obstacles. The secondary task, i.e., the desired hip motion is primarily responsible for the dynamic balance of the robot while doing the task. The effect of such is reflected in the ZMP distribution around the point P_{SZ}. The plot of respective ZMP inside the support polygon projected on $X_W Y_W$-plane is shown in Figure 7g, 7h, and 7i. In all three stages of the task, the ZMP remains well inside the support polygon. The point P_{SZ} is located inside the support polygon at (10, 12.5, 0) cm from the origin of frame $\{G\}$. The tertiary task, i.e., maximizing the manipulability of the primary chain while the robot tries to fulfill the primary task and the secondary task is shown in

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

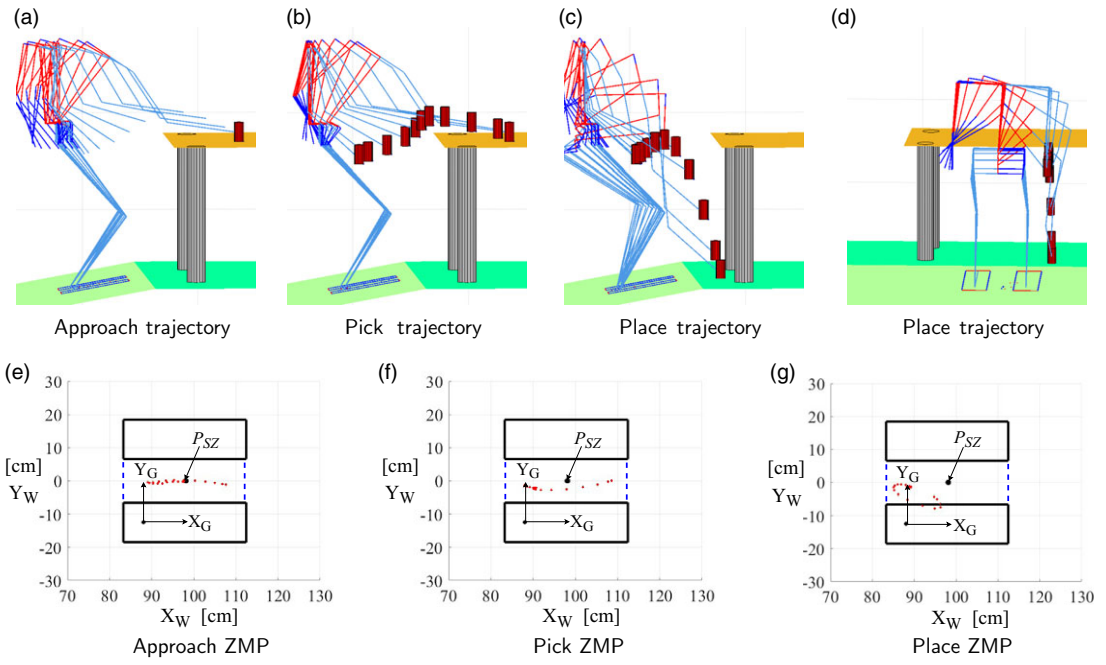


Figure 10. *Two task ($\mathcal{T}_1$ and $\mathcal{T}_2$) decomposition trajectory and ZMP.*

Figure 7j, 7k and 7l. The total work done for all three stages is 179.7 joules. As the task of maximizing the manipulability is of the least priority, manipulability is maximized only in the approach stage and place stage. The surface on which the robot stands has an inclination of $-12.3°$ with respect to $X_W Y_W$-plane along $\hat{Y}_W$ axis. The robot is located at ${}^{W}P_G = \begin{bmatrix}88 & -12.5 & 20\end{bmatrix}^T$ cm and ${}^{G}P_F = \begin{bmatrix}0 & 25 & 0\end{bmatrix}^T$ cm and at an orientation ${}^{W}\boldsymbol{\Theta}_G = \begin{bmatrix}0 & \theta_y & 0\end{bmatrix}^T$ where $\theta_y = -12.3°$ and ${}^{G}\boldsymbol{\Theta}_F = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$. The object is located at $(160, -50, 97)$ cm and destination is at $(130, -22, 31)$ cm. The work is further extended for picking up the object from a cupboard shelf with a 20 cm gap for different object's locations and destination's locations as shown in Figure 11 with ZMP plot in Figure 12. Trajectory and ZMP plot for the slope of the inclined plane of $\theta_y = 5°$ is shown in Figure 13.

5.3. Case II: two task decomposition $\mathcal{T}_1$ and $\mathcal{T}_3$

To execute only subtasks $\mathcal{T}_1$ and $\mathcal{T}_3$, the Eq. (3) is modified as Eq. (24) and k_{opt} is written as $k_{opt} = \begin{bmatrix}k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_6 & k_6\end{bmatrix}^T$ containing six independent variables. So, the number of optimization variables *nvars* is 12.

$$\dot{q}_d = J_1^{\dagger}\dot{r}_1 + (I_8 - J_1^{\dagger}J_1)k \tag{24}$$

Figure 9 is the case of subtask $\mathcal{T}_1$ and subtask $\mathcal{T}_3$. Total energy consumed is 215 joules, which is higher than that of Case I Section 5.2.

5.4. Case III: two task decomposition $\mathcal{T}_1$ and $\mathcal{T}_2$

The third term in Eq. (3) vanishes for subtasks $\mathcal{T}_1$ and $\mathcal{T}_2$, and number of optimization variables *nvars* is 12. The robot was able to reach the object without losing balance; however, it was not able to place the object on the floor without self-collision as there are not enough variables to impose constraints on the body configurations Figure 10. An undesirable motion pattern is seen in Figure 10d.

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

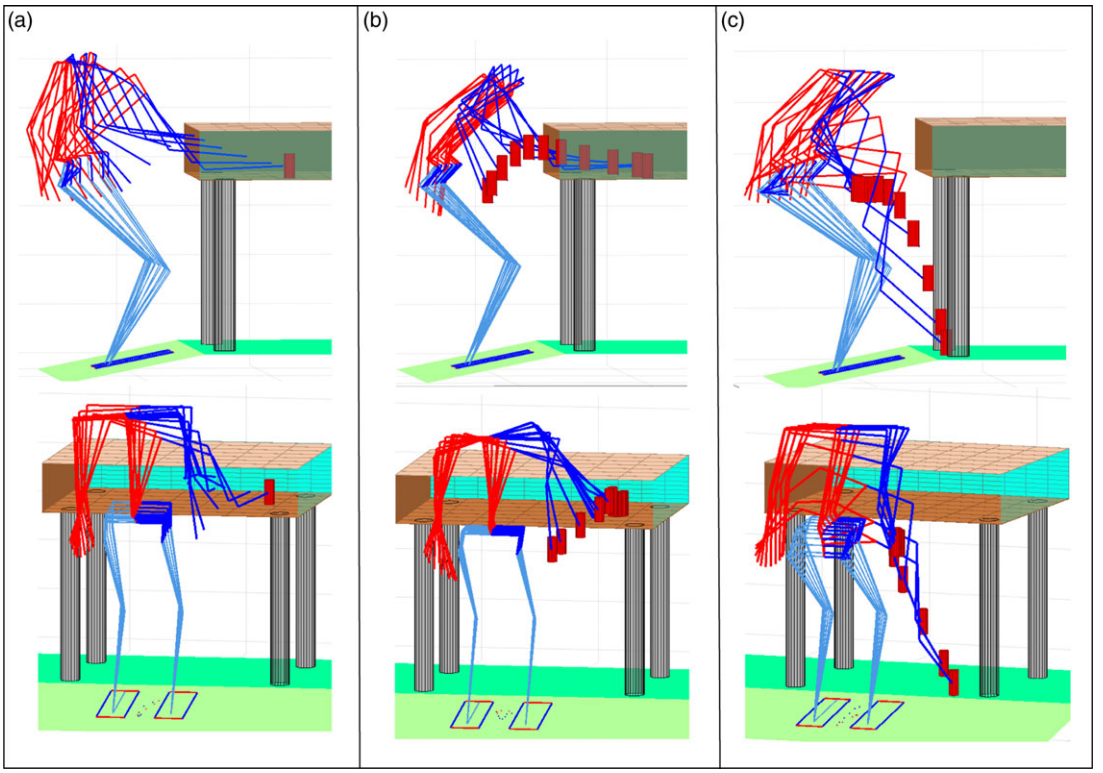


Figure 11. *Trajectories for the three different stages of the task from a shelf of 20 cm gap. (a) Stage I: Approaching the object located at (155, −40, 97) cm. (b) Stage II: Picking up the object from the shelf located at (155, −40, 97) cm. (c) Stage III: Placing the object on the floor at (125, −35, 27) cm.*

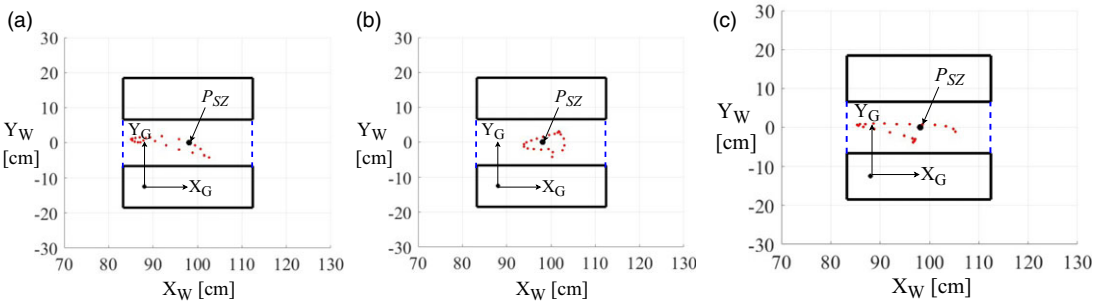


Figure 12. *ZMP plot for pick and place from a shelf. (a) Approaching the object located at (155, −40, 97) cm. (b) Picking up the object from the shelf located at (155, −40, 97) cm. (c) Placing the object on the floor at (125, −35, 27) cm.*

6. Conclusion

In this study, a combination of the Pseudoinverse method of the Jacobian matrix for redundancy resolution and PSO is used to determine an energy-efficient, dynamically balanced trajectory without any collisions with the environment for the pick and place of an object by a humanoid robot standing on inclined ground. Balance is maintained on the inclined ground using ZMP criteria. The key contribution lies in the minimum number of task decomposition and the integration of the entire body dynamics of the humanoid robot (a tree-type robot) to solve the pick and place problem. The robot's entire task is broken down into three distinct subtasks which are optimized using PSO. Comparisons were made

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

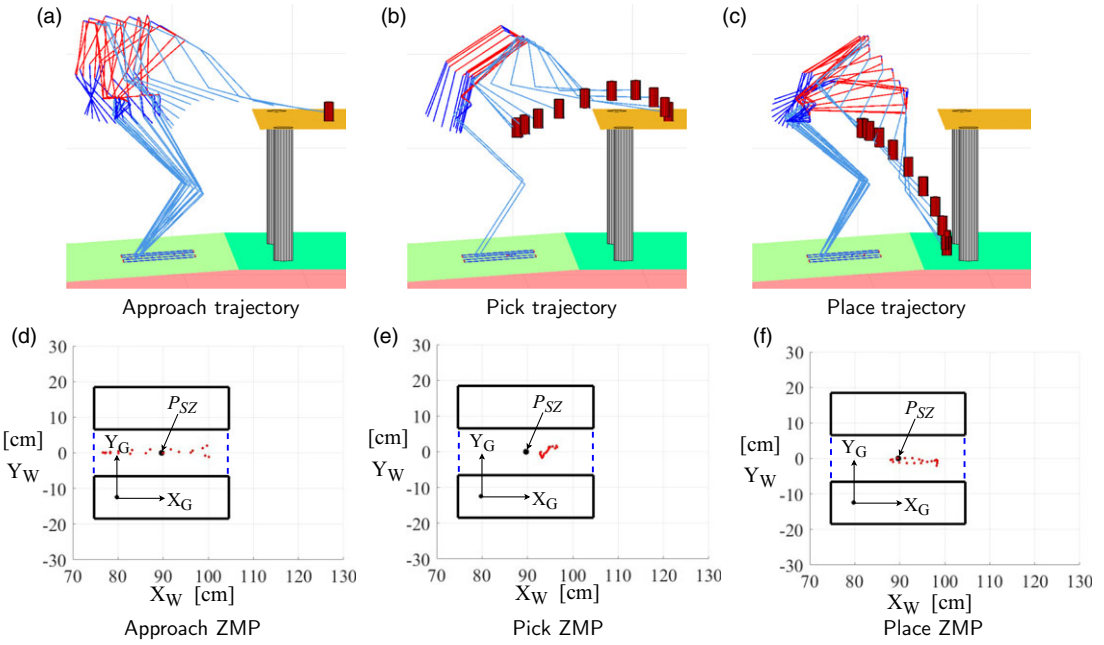


Figure 13. *Trajectory and ZMP plot for slope of inclined surface* $\theta_y = 5°$.

with two task decomposition, and results show that the robot consumed more energy or had undesirable motion when only two tasks were considered. We can conclude that in the case of humanoid robots, a minimum of three task decomposition is required for pick and place application. Increasing the number of task decomposition will still be able to solve the problem. The fourth task could be joint limits on the body of the robot, which is not discussed in our work. In our case, to reduce complexity, the joint limit on the body is set as a constraint in the optimization algorithm. The result shows that the robot can pick and place an object using an energy-optimal trajectory from a table and an enclosed space.

Author contributions. Both the authors James Sorokhaibam and Dr. Ashish Dutta contributed to this research. James Sorokhaibam contributed to the design of the complete algorithm (method), the dynamics of the tree-type robot, implementation, and the simulation. Dr. Ashish Dutta contributed to the trajectory planning, task decomposition, and reviewing and editing the whole paper.

Financial support. This research received no specific grant from any funding agency, commercial, or not-for-profit sectors.

Competing interests. The authors declare no conflicts of interest exist.

Ethical approval. Not applicable under the heading.

References

[1] J. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Auton. Robot* **12**(1), 105–118 (2002).

[2] S. Dalibard, A. Nakhaei, F. Lamiraux and J.-P. Laumond, "Whole-body task planning for a humanoid robot: a way to integrate collision avoidance," article-9th IEEE-RAS International Conference on Humanoid Robots, (Paris, France, IEEE, 2009) pp. 355–360.

[3] K. H. Teja, F. James and S. V. Shah, "Optimal whole body motion planning of humanoid with articulated spine for object manipulation in double support phase," Proceedings of the 2015 Conference on Advances in Robotics, (Goa, India, ACM, 2015) pp. 1–6.

[4] H. T. K., A. Balachandran and S. V. Shah, "Optimal whole-body motion planning of humanoids in cluttered environments," *Robot. Auton. Syst.* **118**, 263–277 (2019). doi:10.1016/j.robot.2019.04.004.

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press

[5] M. Ruchanurucks, "Humanoid robot upper body motion generation using B-spline-based functions," *Robotica* **33**(4), 705–720 (2015).
[6] L. Yu, S. Zhou and S. Huang, "Trajectory optimization of the redundant manipulator with local variable period under multi-machine coordination," *Robotica* **41**(1), 292–305 (2023).
[7] Y. Nakamura, H. Hanafusa and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *Int. J. Rob. Res.* **6**(2), 3–15 (1987).
[8] C. Ott, A. Dietrich and A. Albu-Schäffer, "Prioritized multi-task compliance control of redundant manipulators," *Automatica* **53**, 416–423 (2015).
[9] A. Gams, J. Van den Kieboom, F. Dzeladini, A. Ude and A. J. Ijspeert, "Real-time full body motion imitation on the COMAN humanoid robot," *Robotica* **33**(5), 1049–1061 (2015).
[10] M. Karimi and M. Ahmadi, "A reinforcement learning approach in assignment of task priorities in kinematic control of redundant robots," *IEEE Robot. Autom. Lett.* **7**(2), 850–857 (2021).
[11] Z. Liu, Y. Huang, D. Liu, X. Guo, K. Wang and J. Tan, "Trajectory planning of large redundant manipulator considering kinematic constraints and energy efficiency," *Robotica* **41**(11), 3524–3540 (2023).
[12] M. Tomić, C. Chevallereau, K. Jovanović, V. Potkonjak and A. Rodić, "Human to humanoid motion conversion for dual-arm manipulation tasks," *Robotica* **36**(8), 1167–1187 (2018).
[13] Y. Quan, K. Wang, C. Zhao, C. Lv, H. Zhao and H. Lv, "Obstacle avoidance method for fixed trajectory of a seven-degree-of-freedom manipulator," *Robotica* **41**(5), 1515–1535 (2023).
[14] P. T. Zacharia and E. K. Xidias, "Optimal robot task scheduling in cluttered environments considering mechanical advantage," *Robotica* **42**(9), 1–17 (2024).
[15] J. Zhao and Y. Wei, "A novel algorithm of human-like motion planning for robotic arms," *Int. J. HR* **14**(01), 1650023 (2017).
[16] G. B. Hammam, P. M. Wensing, B. Dariush and D. E. Orin, "Kinodynamically consistent motion retargeting for humanoids," *Int. J. HR* **12**(04), 1550017 (2015).
[17] F. Zhang, J. Qu, H. Liu and Y. Fu, "A multi-priority control of asymmetric coordination for redundant dual-arm robot," *Int. J. HR* **16**(02), 1950008 (2019).
[18] J. W. Jeong and P. H. Chang, "A task-priority based framework for multiple tasks in highly redundant robots," 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, (St. Louis, MO, USA, IEEE, 2009) pp. 5886–5891.
[19] S. Tsuichihara, Y. Hakamata, G. A. G. Ricardez, J. Takamatsu and T. Ogasawara, "Accelerating whole-body motion generation using regression of the torso posture of a humanoid robot," 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), (Cancun, Mexico, IEEE, 2016) pp. 16–21.
[20] L. Sentis and O. Khatib, "A whole-body control framework for humanoids operating in human environments," Proceedings 2006 IEEE International Conference on Robotics and Automation, (Orlando, FL, USA, IEEE, 2006) pp. 2641–2648.
[21] C. Acar and T. Murakami, "Multi-task control for dynamically balanced two-wheeled mobile manipulator through task-priority," 2011 IEEE International Symposium on Industrial Electronics, (Gdansk, Poland, IEEE, 2011) pp. 2195–2200.
[22] S. Tsuichihara, A. Yamaguchi, J. Takamatsu and T. Ogasawara, "Using a weighted pseudo-inverse matrix to generate upper body motion for a humanoid robot doing household tasks," 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), (Zhuhai, China, IEEE, 2015) pp. 333–338.
[23] H. Sugiura, M. Gienger, H. JANßEN and C. Goerick, "Reactive self collision avoidance with dynamic task prioritization for humanoid robots," *Int. J. HR* **7**(01), 31–54 (2010).
[24] P. Kang, H. Meng and W. Xu, "A unified modeling and trajectory planning method based on system manipulability for the operation process of the legged locomotion manipulation system," *Robotica* **41**(7), 2139–2154 (2023).
[25] M. Vukobratović and B. Borovac, "Zero-moment point—thirty five years of its life," *Int. J. HR* **1**(01), 157–173 (2004).
[26] T. Yoshikawa. *Foundations of Robotics: Analysis and Control* (MIT Press, Cambridge, Massachusetts, 1990).
[27] D. E. Witney, "The mathematics of coordinated control of prosthetic arms and manipulators," *J. Dyn. Sys., Meas., Control* **94**(4), 303–309 (1972).
[28] J. J. Craig, "Introduction to robotics: Mechanics and control", (3/E. Pearson EducationIndia, 2009).
[29] P. Sardain and G. Bessonnet, "Forces acting on a biped robot. center of pressure-zero moment point," *IEEE Trans. Syst. Man Cybern. A Syst. Hum.* **34**(5), 630–637 (2004).

Cite this article: J. Sorokhaibam and A. Dutta (2025). "Task priority-based redundancy resolution of a 20 DoFs humanoid robot for object pick and place with minimum energy and dynamic balance", Robotica **43**, 1394–1410. https://doi.org/10.1017/S0263574725000347

https://doi.org/10.1017/S0263574725000347 Published online by Cambridge University Press