ARCHIVE OF MECHANICAL ENGINEERING

DOI: 10.24425/ame.2022.140423 2022, Vol. 69, No. 3, pp. 393–409

Kacper MIKOŁAJCZYK[1], Maksymilian SZUMOWSKI[1], Łukasz WOLIŃSKI[1]

Inverse kinematics solution for humanoid robot minimizing gravity-related joint torques

Received 31 January 2022, **Revised** 19 March 2022, **Accepted** 21 April 2022, **Published online** 1 June 2022

Keywords: humanoid robot, redundant kinematics, joint torque minimization

A method of solving the inverse kinematics problem for a humanoid robot modeled as a tree-shaped manipulator is presented. Robot trajectory consists of a set of trajectories of the characteristic points (the robot's center of mass, origins of feet and hands frames) in the discrete time domain. The description of motion in the frame associated with the supporting foot allows one to represent the robot as a composite of several serial open-loop redundant manipulators. Stability during the motion is provided by the trajectory of the robot's center of mass which ensures that the zero moment point criterion is fulfilled. Inverse kinematics solution is performed offline using the redundancy resolution at the velocity level. The proposed method utilizes robot's redundancy to fulfill joint position limits and to reduce gravity-related joint torques. The method have been tested in simulations and experiments on a humanoid robot Melson, and results are presented.

1. Introduction

In recent times, humanoid robots have become more and more popular [1–5]. They offer unique beneficial capabilities. One of the advantages of humanoid robots is their legged locomotion which makes them superior to their wheeled counterparts when traversing the uneven or unpredictable terrain. As evidenced by the DARPA Robotics Challenge [6], humanoid robots can climb stairs or step over obstacles on the ground. Other tasks of the challenge—simulating unstructured human-suited environment—included moving through tight spaces such as doorways,

✉ Maksymilian Szumowski, email: mszumowski@meil.pw.edu.pl

[1]Faculty of Power and Aeronautical Engineering, Warsaw University of Technology, Warsaw, Poland. ORCIDs: M.S.: 0000-0002-9003-3724; Ł.W.: 0000-0001-9617-4117

maneuvering without losing balance on debris, and utilizing tools and objects devised for human use (power tools and vehicles).

On the other hand, the DARPA Robotics Challenge also proved that with great capabilities of humanoid robots comes great responsibility. Locomotion of legged robots is significantly more convoluted to synthesize than its wheeled counterpart. Footpath planning including proper shaping of the posture is necessary to keep balance. Even though wheeled robots are usually statically stable, the legged robots (notably bipeds) require methods to ensure dynamic stability (e.g. with accordance to commonly used stability criterion—Zero Moment Point Criterion, ZMP [7]).

Furthermore, the complicated structures of humanoid robots give them the ability to freely move their arms and legs. However, the kinematic equations have to allow for the movement of robot's upper and lower limbs and the position of its center of mass. Usually, robot extremities have redundant structure. Therefore, methods for solving inverse kinematics (IK) of redundant manipulators are often used [8]. One possible approach is to compute IK solutions offline and store them in a look-up table for quasi-static walking trajectories [9], generate poses and join them using RRT-Connect method to create smooth motion [10]. Most methods, however, generate set points for each control step [11–13].

Taking advantage of redundancy, a composite kinematic structure has the ability to enhance total performance of the robot as it allows it to perform multiple tasks [11, 14–16]. Kinematic redundancy is often used to avoid obstacles [17–22], but it has also been exploited in a service robot performing the door opening task [23], in a head rising snake robot to track its trajectory [24, 25], and even in a social robot to convey emotions to humans [26].

Given the capabilities of redundant kinematic structures, our motivation for this work was to improve the generated joint trajectories by ensuring that joint limits are fulfilled and driving torques in joints are minimized. Joint torques minimization has plenty of benefits, especially for a legged robot. First of all, the smaller the joint torques the smaller the energy consumption and longer working time on single battery charge. Secondly, smaller torque coincides with smaller motor current, which in turn generates less heat in actuators. Lastly, reduced torque demand allows for the design of lighter servo-motors resulting in a robot with improved agility.

The aim of this work is to develop the offline method of solving inverse kinematics of a redundant robot tracking a desired task space trajectory which fulfills the ZMP criterion. At the same time, the redundancy resolution should reduce the proposed cost function (joint limit avoidance and joint torques minimization). The outcomes of the adoption of different cost functions have been validated on a humanoid robot *Melson* designed by the authors and are presented in this article. This article extends authors' previous work [27].

This paper is organized as follows: section 2 presents the problem of motion planning for humanoid robots, section 3 describes the theoretical foundations of

inverse kinematics of redundant tree-shaped robots which are then applied to a humanoid robot in section 4. Section 5 describes the results of simulations and experiments, while section 6 concludes the paper.

2. Problem formulation

As mentioned in the Introduction, planning a motion of a humanoid robot is a challenging task. Here, we use the Motion Generation System (MGS). This system was developed and tested on our previous platform [28, 29]. Fig. 1 presents the structure and data flow of four main modules:

1. Motion Planner
2. Trajectories Planner
3. Center of Mass (CoM) Trajectory Generator
4. Inverse Kinematics Solver

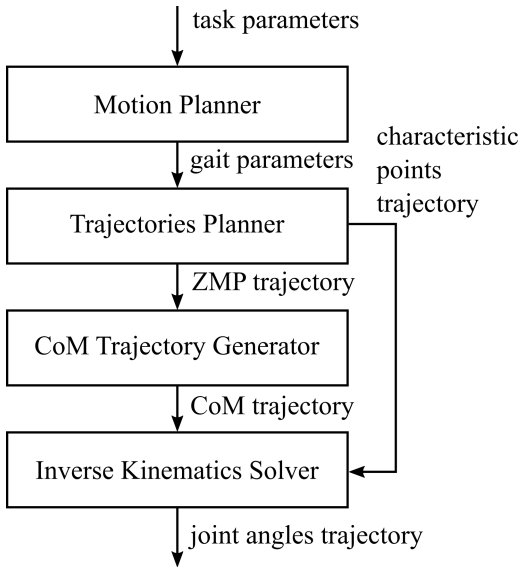

Fig. 1. Data flow in Motion Generation System

The role of the Motion Planner is to generate set of foot placements on the ground surface for the given task specified by its parameters. Currently—as Melson was designed with RoboCup Soccer tournament in mind—the module allows for planning the following tasks: walking, kicking the ball, and picking up an object [30]. The output of the Motion Planner is the set of gait parameters.

The gait parameters are fed into the Trajectories Planner module which generates the desired trajectory of the Zero Moment Point (ZMP) and of the characteristic points on the robot. The trajectory of the characteristic points include the feet trajectory for each transfer phase and also hands trajectory for whole motion. More

details about the feet and hands motion, and the transfer phases can be found in [29, 30].

Using the ZMP trajectory output by the previous module, the Center of Mass (CoM) Trajectory Generator computes the trajectory of the robot's center of mass which satisfies the Zero Moment Point criterion. As a solution for CoM generation algorithm, the module utilizes the Preview Control method similarly as in [31]. More details about the implementation of the Preview Control can be found in [28].

The last module, called the Inverse Kinematics Solver, takes as an input the CoM trajectory and trajectory of the characteristic points and computes the joint angles trajectory. The computed joint angles trajectory is sent to the robot and realized in an open-loop control strategy.

In this paper we will focus on the walking task of the humanoid robot and on the last module of the Motion Generation System—the Inverse Kinematics Solver. Unfortunately, an analytical solution of the inverse kinematics problem is not practical. However, when the humanoid robot stands with one foot on the ground, it can be treated as a tree-like manipulator with a redundant kinematic structure. Therefore, to compute the joint angles, the methods of solving the inverse kinematics for redundant manipulators can be used.

The main goal of this work is to utilize the redundancy resolution to generate such a motion for a humanoid robot which will avoid the angle limits in joints and minimize the gravity-related joint torques. The details of the proposed method will be elaborated in section 3, and its application to Melson robot will be described in section 4.

3. Inverse kinematics of redundant tree-shaped robots

3.1. Preliminaries

In a tree-like kinematic structure, task coordinates—position and/or orientation—of the i-th branch end effector are stored in the vector $\mathbf{t}_i^{\mathrm{EE}}$ given as [32]:

$$\mathbf{t}_i^{\mathrm{EE}} = \mathbf{f}_i(\mathbf{q}), \tag{1}$$

where $i = 1, 2, \ldots, p$, and p is the number of branches, $\mathbf{q} \in \mathbb{R}^n$ is the vector of joint coordinates, n is the number of degrees of freedom of the whole tree, $\mathbf{f}_i(\mathbf{q})$ is the forward kinematics function of the i-th branch, and $\mathbf{t}_i^{\mathrm{EE}} \in \mathbb{R}^{m_i}$ where m_i is the number of the task coordinates of the i-th branch end effector.

Therefore, the vector $\mathbf{t}$ of task coordinates of all end effectors can be written as:

$$\mathbf{t} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{EE}T} & \ldots & \mathbf{t}_p^{\mathrm{EE}T} \end{bmatrix}^T = \begin{bmatrix} \mathbf{f}_1^T(\mathbf{q}) & \ldots & \mathbf{f}_p^T(\mathbf{q}) \end{bmatrix}^T = \mathbf{f}(\mathbf{q}), \tag{2}$$

and $\mathbf{t}, \mathbf{f}(\mathbf{q}) \in \mathbb{R}^m$ where m is:

$$m = \sum_{i=1}^{p} m_i. \tag{3}$$

Differentiating (2) with respect to time results in the following equation:

$$\dot{\mathbf{t}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{4}$$

which illustrates the relationship between the time derivative of the task coordinates vector $\dot{\mathbf{t}}$ and the joint velocities vector $\dot{\mathbf{q}}$. The Jacobian matrix $\mathbf{J} \in \mathbb{R}^{m \times n}$ is computed as [12, 13, 32]:

$$\mathbf{J} = \frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}. \tag{5}$$

Solving Eq. (4) is usually done by inverting the Jacobian matrix [12, 13, 33]:

$$\dot{\mathbf{q}} = \mathbf{J}^{+}\dot{\mathbf{t}} + \mathbf{P}\dot{\mathbf{q}}^{0}, \tag{6}$$

where $\mathbf{J}^{+} \in \mathbb{R}^{n \times m}$ is the pseudoinverse of the Jacobian matrix, $\dot{\mathbf{q}}^{0} \in \mathbb{R}^{n}$ is the arbitrarily chosen vector of joint velocities to be projected on the null space of $\mathbf{J}$ (see section 3.2), and $\mathbf{P} \in \mathbb{R}^{n \times n}$ is the null space projection matrix, usually given as [12, 13]:

$$\mathbf{P} = \mathbf{I} - \mathbf{J}^{+}\mathbf{J}, \tag{7}$$

where $\mathbf{I}$ is the identity matrix of the appropriate size.

The robot is controlled in the discrete time domain. Therefore, joint positions to be sent to the controller are obtained by using the explicit Euler integration method:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \dot{\mathbf{q}}_k \Delta t, \tag{8}$$

where Δt is the time step length, $\mathbf{q}_k = \mathbf{q}(k\Delta t)$ is the vector of the joint positions at the k-th time step, and $\dot{\mathbf{q}}_k = \dot{\mathbf{q}}(k\Delta t)$ is the vector of the joint velocities at the k-th time step.

To avoid the drift associated with the numerical integration, the closed-loop inverse kinematics (CLIK) method [12, 13, 32, 34, 35] is used to calculate the joint velocities $\dot{\mathbf{q}}_k$ in (8):

$$\dot{\mathbf{q}}_k = \mathbf{J}_k^{+}\left(\frac{\mathbf{t}_{k+1} - \mathbf{f}(\mathbf{q}_k)}{\Delta t}\right) + \mathbf{P}_k\dot{\mathbf{q}}_k^{0}, \tag{9}$$

where $\dot{\mathbf{q}}_k^{0} = \dot{\mathbf{q}}^{0}(k\Delta t)$ is the vector of the joint velocities to be projected on the null space at the k-th time step, $\mathbf{J}_k = \mathbf{J}(\mathbf{q}_k)$ is the Jacobian matrix at the k-th time step, and $\mathbf{P}_k = \mathbf{P}(\mathbf{q}_k)$ is the null space projection matrix at the k-th time step. The CLIK feedback loop is achieved by approximating the time derivative of the task space coordinates vector $\dot{\mathbf{t}}_k$ with a differential equation:

$$\dot{\mathbf{t}}_k \approx \frac{\mathbf{t}_{k+1} - \mathbf{f}(\mathbf{q}_k)}{\Delta t}, \tag{10}$$

where $\mathbf{t}_{k+1} = \mathbf{t}\,((k+1)\Delta t)$ is the desired task coordinates vector at the $(k+1)$-th step, and forward kinematics $\mathbf{f}(\mathbf{q}_k)$ is calculated for the joint positions $\mathbf{q}_k$ obtained at the k-th step of the solution.

3.2. Redundancy handling

A widely used technique for selecting the vector $\dot{\mathbf{q}}^0$ in (6) is the gradient projection method [12, 13, 36, 37]:

$$\dot{\mathbf{q}}^0 = -\gamma \nabla H(\mathbf{q}), \tag{11}$$

where γ is a scalar gain, and $H(\mathbf{q})$ is the cost function to minimize.

The gradient projection method can be utilized to keep the joint positions away from their limits by using the cost function $H(\mathbf{q})$ defined as [12, 13, 38]:

$$H_{\text{joint}}(\mathbf{q}) = \frac{1}{2} \sum_{j=1}^{n} \left(\frac{q_j - q_{j,\text{mid}}}{q_{j,\max} - q_{j,\min}} \right)^2, \tag{12}$$

where $q_{j,\max}$ and $q_{j,\min}$ are respectively the upper and lower bound on the j-th joint position, and $q_{j,\text{mid}} = \frac{1}{2}\left(q_{j,\max} + q_{j,\min}\right)$. Additional bonus of this function is that it guaranties cyclic joint motions, avoiding the joint drift.

Another possibility is to use the cost function $H(\mathbf{q})$ to minimize the gravity-related joint torques:

$$H_{\text{grav}}(\mathbf{q}) = \frac{1}{2} \boldsymbol{\tau}^T \mathbf{W} \boldsymbol{\tau}, \tag{13}$$

where $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of gravity-related torques in joints, and $\mathbf{W} \in \mathbb{R}^{n \times n}$ is the diagonal positive-definite gain matrix which allows us to define the magnitude of minimization of each component in $\boldsymbol{\tau}$ separately.

We propose to combine (12) and (13) together to achieve two goals—joint limits avoidance and joint torques minimization—at the same time. In this framework, we rewrite Eq. (11) in the following form:

$$\dot{\mathbf{q}}^0 = -\gamma_{\text{joint}} \nabla H_{\text{joint}}(\mathbf{q}) - \gamma_{\text{grav}} \nabla H_{\text{grav}}(\mathbf{q}). \tag{14}$$

3.3. Gravity-related joint torques

Gravity-related joint torques $\boldsymbol{\tau}$ in (13) can be computed using recursive algorithms for dynamic modeling of robotic manipulators [39–42]. In this article, we utilize the Lagrange formulation, similarly as in [12, 43]. The potential energy U of the robot is computed as:

$$U = -\sum_{j=1}^{n} m_j \mathbf{g}^T \mathbf{r}_{\text{CoM,j}}, \tag{15}$$

where m_j is the mass of the j-th link, $\mathbf{g} = \begin{bmatrix} 0 & 0 & -9.81 \end{bmatrix}^T$ is the vector of the gravitational acceleration expressed in base frame, and $\mathbf{r}_{\text{CoM,j}}$ is the position of j-th

link center of mass in base frame. The gravity-related torque in the joint i is then computed as [43]:

$$\tau_i = -\sum_{j=1}^{n} m_j \mathbf{g}^T \frac{\partial \mathbf{r}_{\mathrm{CoM,j}}}{\partial q_i} = -\sum_{j=1}^{n} m_j \mathbf{g}^T \mathbf{J}_{\mathbf{v}_i}^{(l_j)}(\mathbf{q}), \tag{16}$$

for $i = 1, \ldots n$. The vector $\mathbf{J}_{\mathbf{v}_i}^{(l_j)} \in \mathbb{R}^3$ is an element of the geometric Jacobian matrix (not to be confused with the task Jacobian (5)) and is given by the following expression [43]:

$$\mathbf{J}_{\mathbf{v}_i}^{(l_j)} = \mathbf{z}_{i-1} \times \left(\mathbf{r}_{\mathrm{CoM,j}} - \mathbf{r}_{i-1}\right), \tag{17}$$

where $\mathbf{r}_{i-1}$ is the position of the origin of frame $i-1$, $\mathbf{z}_{i-1}$ is the unit vector of axis z of frame $i-1$ (both expressed in the base frame):

$$\mathbf{z}_{i-1} = \mathbf{R}_{i-1}^{0} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T. \tag{18}$$

However, computing the joint torques using (17) requires n^2 iterations. The numerical burden can be greatly reduced by defining the Jacobian matrix $\mathbf{J}_{\mathrm{CoM}} \in \mathbb{R}^{3\times n}$ which describes the relationship between the joint space velocity and the translational velocity of the robot's center of mass $\mathbf{v}_{\mathrm{CoM}}$ [44]:

$$\mathbf{J}_{\mathrm{CoM}}\dot{\mathbf{q}} = \mathbf{v}_{\mathrm{CoM}}. \tag{19}$$

Each column i of $\mathbf{J}_{\mathrm{CoM}}$ can be computed as:

$$\mathbf{J}_{\mathrm{CoM,i}} = \frac{\partial \mathbf{r}_{\mathrm{CoM}}}{\partial q_i}, \tag{20}$$

where the position of the robot's center of mass $\mathbf{r}_{\mathrm{CoM}}$ is given by:

$$\mathbf{r}_{\mathrm{CoM}} = \frac{1}{M}\sum_{j=1}^{n} m_j \mathbf{r}_{\mathrm{CoM,j}}, \tag{21}$$

where m_j is the mass of j-th link, and $\mathbf{r}_{\mathrm{CoM,j}}$ is the position of the center of mass of j-th link in base frame. Total mass of the entire robot is $M = \sum_{j=1}^{n} m_j$.

Combining equations (20) and (21) gives:

$$\mathbf{J}_{\mathrm{CoM,i}} = \frac{1}{M}\sum_{j=1}^{n} m_j \frac{\partial \mathbf{r}_{\mathrm{CoM,j}}}{\partial q_i} = \frac{1}{M}\sum_{j=1}^{n} m_j \mathbf{J}_{\mathbf{v}_i}^{(l_j)} \tag{22}$$

where $\mathbf{J}_{\mathbf{v}_i}^{(l_j)}$ is an element of the geometric Jacobian matrix introduced in (17). Next, equation (16) can be slightly reorganized:

$$\tau_i = -\mathbf{g}^T \sum_{j=1}^{n} m_j \mathbf{J}_{\mathbf{v}_i}^{(l_j)}, \tag{23}$$

where the sum part can be replaced by using formula (22):

$$\tau_i = -\mathbf{g}^T M \mathbf{J}_{\mathrm{CoM,i}} = -\mathbf{J}_{\mathrm{CoM,i}}^T \cdot M\mathbf{g}. \tag{24}$$

Assuming that the analytical expression for (20) is known, the computation of (24) requires only n iterations.

4. Inverse kinematics of the humanoid robot Melson

4.1. Description of the robot

Melson (pictured in Figs. 6, 7 and 8 in section 5) is a humanoid robot built by the students of the Students' Robotic Association at the Faculty of Power and Aeronautical Engineering of the Warsaw University of Technology. It was designed for Humanoid Sumo and Humanoid Sprint disciplines popular in many robotic competitions and for RoboCup Soccer tournament. Melson has 19 degrees of freedom (DoF), and its anthropomorphic kinematic structure resembles the structure of the human body. It was modeled on an adult human body with the lengths of body parts based on biomechanical data from [45] and scaled down. Overall, Melson is 0.5 m tall and weighs 2.6 kg. Its kinematic structure allows for translating and rotating each foot relative to torso (6 DoFs per leg), setting position of hands (3 DoFs per arm) and twisting the torso (1 DoF). Melson's joints are actuated by 12 Dynamixel RX-28 servos—in the lower limbs—and 7 Dynamixel AX-12—in the upper limbs and torso.

4.2. Application of the inverse kinematics solver to Melson

A graph representing the kinematic structure of Melson standing on its right foot is shown in Fig. 2. Each node represents a body part which consists of one or more links (e.g., a foot consists of two links, a shank of one link, and a thigh of three links). Each link is connected to its neighbor via a revolute joint (actuated by a servomotor), but connections between body parts have more degrees of freedom—just like joints in the human body—and are represented as straight lines between the nodes.

The trajectory computed by the Trajectories Planner and CoM Trajectory Generator modules is output in the form of the following vector for each time step:

$$\mathbf{r}^{(0)} = \begin{bmatrix} \mathbf{r}_{\mathrm{RF}}^{(0)T} & \mathbf{r}_{\mathrm{LF}}^{(0)T} & \mathbf{r}_{\mathrm{RH}}^{(0)T} & \mathbf{r}_{\mathrm{LH}}^{(0)T} & \mathbf{r}_{\mathrm{CoM}}^{(0)\ T} & \boldsymbol{\phi}_{\mathrm{RF}}^{(0)T} & \boldsymbol{\phi}_{\mathrm{LF}}^{(0)T} \end{bmatrix}_{21\times 1}^{T}, \tag{25}$$

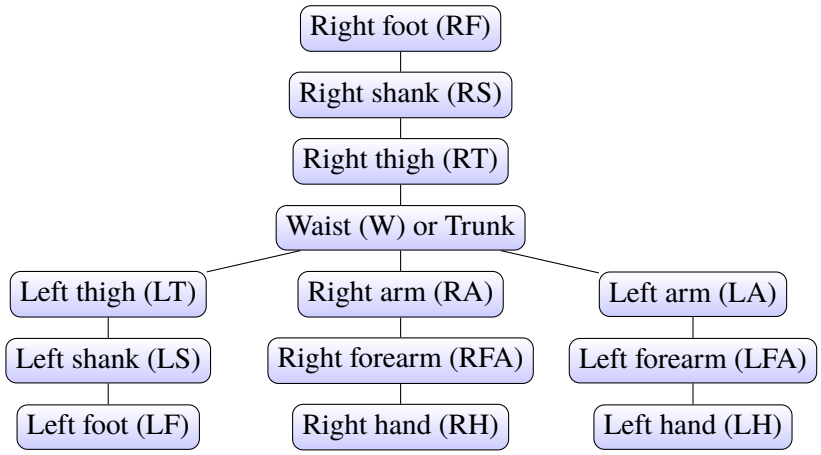

Fig. 2. Representation of Melson's kinematic structure in a tree form

where $\mathbf{r}_{\mathrm{RF}}^{(0)}$ is the position of the origin of the right foot body frame, $\mathbf{r}_{\mathrm{LF}}^{(0)}$ is the position of the origin of the left foot body frame, $\mathbf{r}_{\mathrm{RH}}^{(0)}$ is the position of the origin of the right hand body frame $\mathbf{r}_{\mathrm{LH}}^{(0)}$ is the position of the origin of the left hand body frame $\mathbf{r}_{\mathrm{CoM}}^{(0)}$ is the position of the origin of the robot's center of mass frame, all in the frame associated with the ground. Moreover, $\boldsymbol{\phi}_{\mathrm{RF}}^{(0)}$ and $\boldsymbol{\phi}_{\mathrm{LF}}^{(0)}$ are the orientation of the right and left foot, respectively, in the ground frame, represented by yaw-pitch-roll angles ($z-y-x$ Euler angles).

However, as the robot is walking, it is more convenient to associate the base frame with the supporting foot frame. If the right foot is the base frame, then the following transformations are used:

$$\begin{aligned}
\mathbf{r}_{\mathrm{LF}}^{(\mathrm{RF})} &= \mathbf{R}_{\mathrm{RF}}^{0\,T}\left(\mathbf{r}_{\mathrm{LF}}^{(0)} - \mathbf{r}_{RF}^{(0)}\right)\\
\mathbf{r}_{\mathrm{RH}}^{(\mathrm{RF})} &= \mathbf{R}_{\mathrm{RF}}^{0\,T}\left(\mathbf{r}_{\mathrm{RH}}^{(0)} - \mathbf{r}_{RF}^{(0)}\right)\\
\mathbf{r}_{\mathrm{LH}}^{(\mathrm{RF})} &= \mathbf{R}_{\mathrm{RF}}^{0\,T}\left(\mathbf{r}_{\mathrm{LH}}^{(0)} - \mathbf{r}_{RF}^{(0)}\right)\\
\mathbf{r}_{\mathrm{CoM}}^{(\mathrm{RF})} &= \mathbf{R}_{\mathrm{RF}}^{0\,T}\left(\mathbf{r}_{\mathrm{CoM}}^{(0)} - \mathbf{r}_{\mathrm{RF}}^{(0)}\right)\\
\mathbf{R}_{\mathrm{LF}}^{\mathrm{RF}} &= \mathbf{R}_{\mathrm{RF}}^{0\,T}\mathbf{R}_{\mathrm{LF}}^{0}
\end{aligned} \tag{26}$$

where $\mathbf{R}_{\mathrm{RF}}^{0}$ is the rotation matrix between the ground frame and the right foot frame calculated using the $\boldsymbol{\phi}_{\mathrm{RF}}^{(0)}$ angles. The rotation matrix $\mathbf{R}_{\mathrm{LF}}^{\mathrm{RF}}$ is used to obtain the orientation $\boldsymbol{\phi}_{\mathrm{LF}}^{(\mathrm{RF})}$ of the left foot in the right foot frame. Using Eq. (26), a set of

task coordinates is obtained:

$$\mathbf{t} = \begin{bmatrix} \mathbf{r}_{\mathrm{LF}} \\ \mathbf{r}_{\mathrm{RH}} \\ \mathbf{r}_{\mathrm{LH}} \\ \mathbf{r}_{\mathrm{CoM}} \\ \boldsymbol{\phi}_{\mathrm{LF}} \end{bmatrix}_{15\times 1}, \tag{27}$$

where the superscript RF is omitted for readability.

Having the task coordinates defined in (27), the Jacobian matrix is computed using (5). The solution to the inverse kinematics of Melson is obtained by the use of (9) with the secondary task (14).

In each single support phase, the joint torques are computed according to (24), starting from the supporting foot (the supporting foot is a base). During the double support phase, the situation is more complicated as the structure of the robot becomes a closed loop chain. Therefore, equation (24) cannot be used directly. To alleviate this problem, we propose to switch off the secondary task given by (13) for the duration of double support phases. In other words, γ_{grav} is constant for much of the single support phase, but quickly goes linearly to zero just before the double support phase starts. It is a simplification, but it allows us to minimize gravity-related joint torques during single support phases. Future works will include expanding our method to cover also the double support phase.

5. Experiments

This section is devoted to the results of applying the Inverse Kinematics Solver to the walking task of Melson. Different values of the parameters γ_{joint} and γ_{grav} in (14) are considered. Three scenarios are tested, each with a different secondary task (defined by the values of γ_{joint} and γ_{grav}):

1. None ($\gamma_{\mathrm{joint}} = 0$ and $\gamma_{\mathrm{grav}} = 0$).
2. Joint limits avoidance ($\gamma_{\mathrm{joint}} \neq 0$ and $\gamma_{\mathrm{grav}} = 0$).
3. Joint limits avoidance combined with minimization of the joint torques ($\gamma_{\mathrm{joint}} \neq 0$ and $\gamma_{\mathrm{grav}} \neq 0$).

Torque and joint trajectories for selected joints are presented in Fig. 3 (hip around x axis), Fig. 4 (knee) and Fig. 5 (ankle around x axis). For the hip at x axis (Fig. 3) we can see that in scenario 3 torque has been significantly reduced. Unfortunately, peaks of position at this joint have increased. For the knee (Fig. 4) we can observe similar behavior. In scenario 3 peak torque has been reduced, but position peak has slightly increased. Interestingly, for the ankle at x axis (Fig. 5) we can observe a different result. In scenario 3 the torque has not been reduced—instead, position peak for those scenario has been reduced.

For each scenario the task space trajectory generated in the MGS is the same. Gait parameters are set to perform 5 cm forward translation of the robot in 12 s

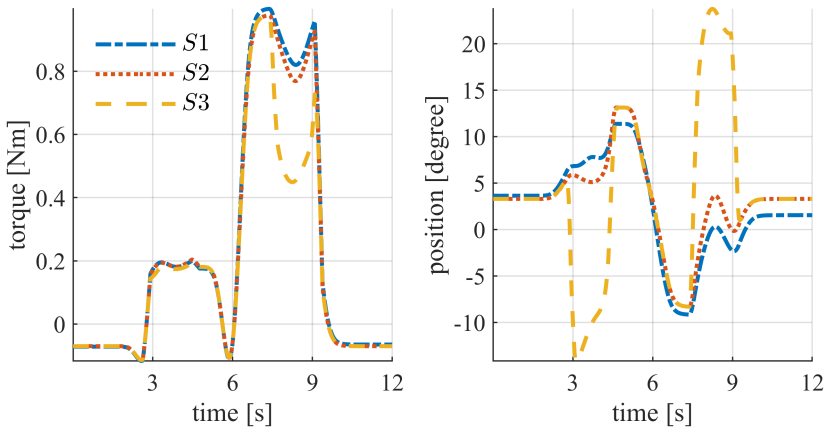

Fig. 3. Joint torque and position for the left leg's hip at x axis. S_1 denotes scenario 1 ($\gamma_{\text{joint}} = 0.0$ and $\gamma_{\text{grav}} = 0.0$), S_2 denotes scenario 2 ($\gamma_{\text{joint}} = 1.0$ and $\gamma_{\text{grav}} = 0.0$) and S_3 denotes scenario 3 ($\gamma_{\text{joint}} = 0.35$ and $\gamma_{\text{grav}} = 1.6$)

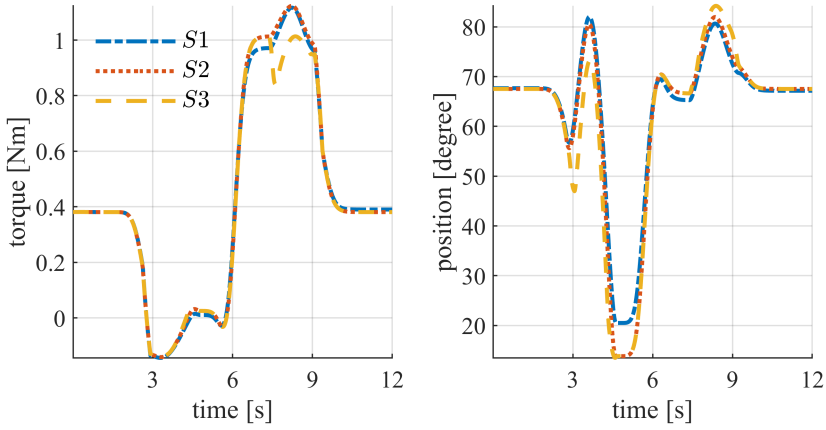

Fig. 4. Joint torque and position for left leg's knee. S_1 denotes scenario 1 ($\gamma_{\text{joint}} = 0.0$ and $\gamma_{\text{grav}} = 0.0$), S_2 denotes scenario 2 ($\gamma_{\text{joint}} = 1.0$ and $\gamma_{\text{grav}} = 0.0$) and S_3 denotes scenario 3 ($\gamma_{\text{joint}} = 0.35$ and $\gamma_{\text{grav}} = 1.6$)

using 2 steps. Sampling time is set to $\Delta t = 0.01$ s. All scenarios are recorded and presented in the movie https://youtu.be/KPiP6PhSQs4. For scenario 1 snapshots of robots motion are presented in Fig. 6. In scenario 2, the values of used parameters are $\gamma_{\text{joint}} = 1.0$ and $\gamma_{\text{grav}} = 0.0$. Snapshots for this scenario are presented in Fig. 7. In scenario 3, the values of used parameters are $\gamma_{\text{joint}} = 0.35$ and $\gamma_{\text{grav}} = 1.6$. Snapshots for this scenario are presented in Fig. 8. The values γ_{joint} and γ_{grav} were selected empirically.

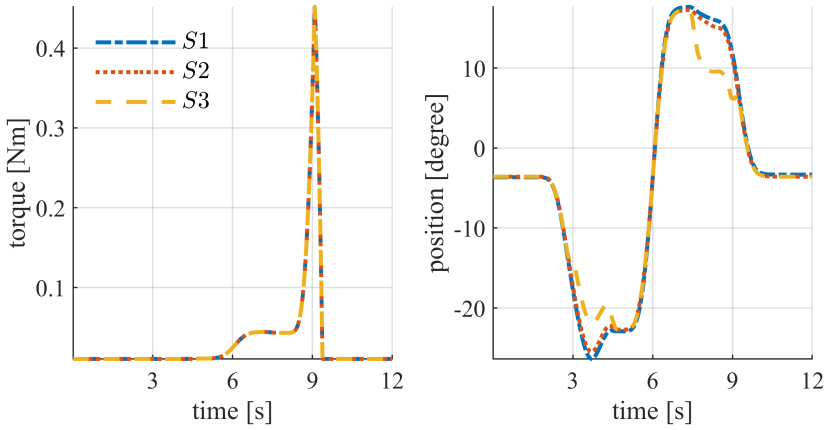

Fig. 5. Joint torque and position for left leg's ankle at x axis. S_1 denotes scenario 1 ($\gamma_{\text{joint}} = 0.0$ and $\gamma_{\text{grav}} = 0.0$), S_2 denotes scenario 2 ($\gamma_{\text{joint}} = 1.0$ and $\gamma_{\text{grav}} = 0.0$) and S_3 denotes scenario 3 ($\gamma_{\text{joint}} = 0.35$ and $\gamma_{\text{grav}} = 1.6$)

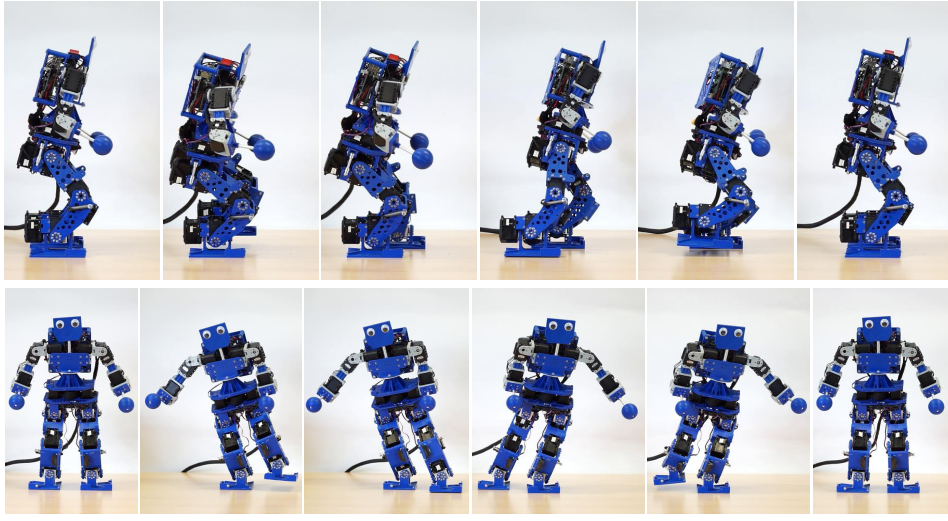

Fig. 6. Melson walking forward—side view (top row) and front view (bottom row). IK cost function parameters: $\gamma_{\text{joint}} = 0.00$ and $\gamma_{\text{grav}} = 0.0$

For the final test we wanted to highlight the performance of the Inverse Kinematics Solver in the context of long and periodical movement. For this test we have set the long walk of 30 steps. Only scenario 1 (no secondary task) and 2 (joint limits avoidance with $\gamma_{\text{joint}} = 1.0$) are considered. Start and final pose for both scenarios are presented in Fig. 9. In this figure we can see that final pose of the robot after 30 steps is different for different values of parameter γ_{joint}. Worth noting is the fact that in both cases the task trajectory is performed correctly. The only difference is in the

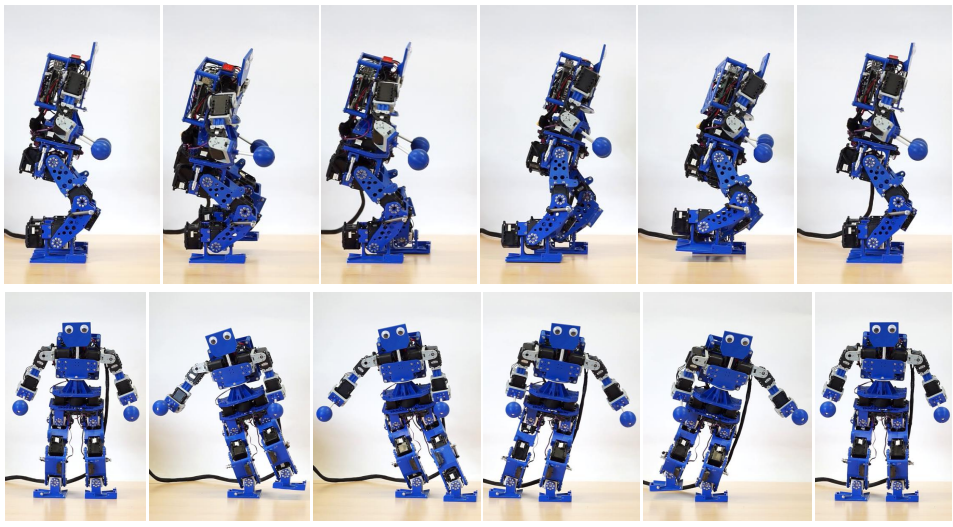

Fig. 7. Melson walking forward—side view (top row) and front view (bottom row). IK cost function parameters: $\gamma_{\text{joint}} = 1.0$ and $\gamma_{\text{grav}} = 0.0$

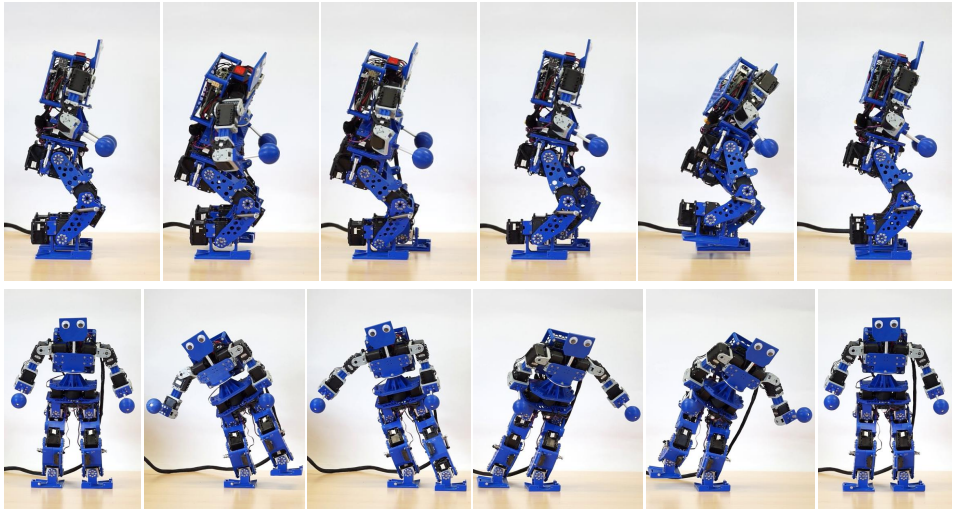

Fig. 8. Melson walking forward—side view (top row) and front view (bottom row). IK cost function parameters: $\gamma_{\text{joint}} = 0.35$ and $\gamma_{\text{grav}} = 1.6$

joint trajectory which is illustrated by Fig. 10. This figure presents the right leg hip at x axis trajectory for both scenarios. In the first scenario (parameter $\gamma_{\text{joint}} = 0.0$) the joint drift is visible, while for the second scenario (parameter $\gamma_{\text{joint}} = 1.0$) the joint drift is not present.

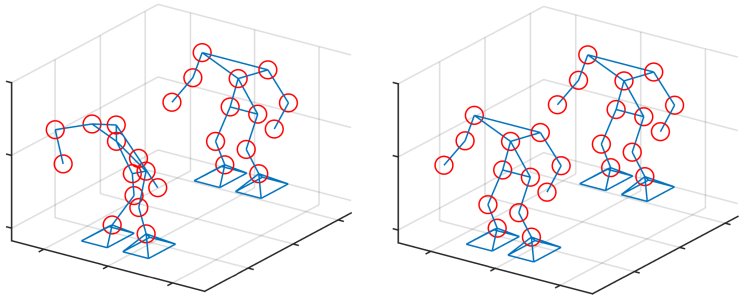

Fig. 9. Robot's beginning and final pose after 30 steps. Left picture IK Solver parameter $\gamma_{\text{joint}} = 0.0$, while right picture is generated with value of parameter $\gamma_{\text{joint}} = 1.0$

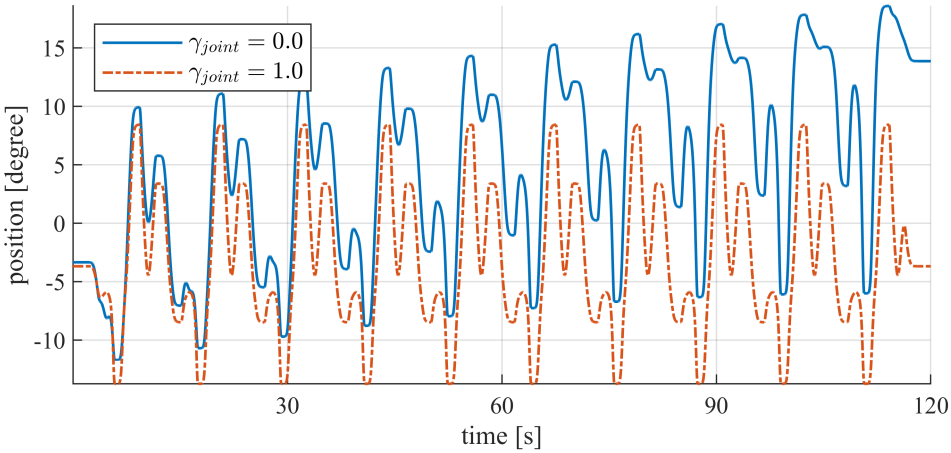

Fig. 10. Joint drift in right leg hip at x axis

6. Conclusions

This paper presents the description, implementation and validation of the Inverse Kinematics Solution for redundant tree-shaped kinematics of the humanoid robot Melson.

Redundancy resolution is performed via the null space projection of selected joint velocity vector. For calculating this joint velocity vector we use gradient projection method with two different cost functions. First function keeps the joint positions away from their limits. Second is a combination of this goal with the minimization of the gravity-related joint torques.

For comparative analysis we have presented experiments of three different scenarios. One without utilization of the null space projection and two experiments with different secondary tasks projected onto the null space of the robot's Jacobian matrix. Currently, our method permits to reduce gravity-related joint torques during

single support phases. Future works will include expanding our method to cover also the double support phase.

The Inverse Kinematics Solution module proved to be versatile enough to test various scenarios. All presented experiments validated correctness of the proposed algorithm. Final experiment showed that the combination of the two different goals is possible and provides results that improve motion of the humanoid robot.

References

[1] P. Gupta, V. Tirth, and R.K. Srivastava. Futuristic humanoid robots: An overview. In *First International Conference on Industrial and Information Systems*, pages 247–254, 2006. doi: 10.1109/ICIIS.2006.365732.

[2] S. Behnke. Humanoid robots – from fiction to reality? *KI– Künstliche Intelligenz*, 22(4):5–9, 2008.

[3] C.-H. Ting, W.-H Yeo, Y.-J. King, Y.-D. Chuah, J.-V Lee, and W.-B Khaw. Humanoid robot: A review of the architecture, applications and future trend. *Research Journal of Applied Sciences, Engineering and Technology*, 7:1178–1183, 2014. doi: 10.19026/rjaset.7.402.

[4] R. Mahum, F. Butt, K. Ayyub, S. Islam, M. Nawaz, and D. Abdullah. A review on humanoid robots. *International Journal of Advanced and Applied Sciences*, 4(2):83–90, 2017. doi: 10.21833/ijaas.2017.02.015.

[5] S. Saeedvand, M. Jafari, H.S. Aghdasi, and J. Baltes. A comprehensive survey on humanoid robot development. *The Knowledge Engineering Review*, 34:e20, 2019. doi: 10.1017/S0269888919000158.

[6] E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski. The DARPA robotics challenge finals: Results and perspectives. *Journal of Field Robotics*, 34(2):229–240, 2016. doi: 10.1002/rob.21683.

[7] M. Vukobratović and B. Borovac. Zero-Moment Point — Thirty Five Years of Its Life. *International Journal of Humanoid Robotics*, 01(01):157–173, 2004. doi: 10.1142/s0219843604000083.

[8] Ł. Woliński and M. Wojtyra. A novel QP-based kinematic redundancy resolution method with joint constraints satisfaction. *IEEE Access*, 10:41023–41037, 2022. doi: 10.1109/ACCESS.2022.3167403.

[9] B.W. Satzinger, J.I. Reid, M. Bajracharya, P. Hebert, and K. Byl. More solutions means more problems: Resolving kinematic redundancy in robot locomotion on complex terrain. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4861–4867. IEEE, 2014. doi: 10.1109/iros.2014.6943253.

[10] J.J. Kuffner and S.M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. IEEE, 2000. doi: 10.1109/robot.2000.844730.

[11] B. Siciliano and J.J.E. Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 1211–1216, vol. 2, 1991. doi: 10.1109/ICAR.1991.240390.

[12] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics. Modelling, Planning and Control*. Springer-Verlag, Wien, 1 edition, 2009. doi: 10.1007/978-1-84628-642-1.

[13] B. Siciliano and O. Khatib, editors. *Springer Handbook of Robotics*. Springer Handbooks. Springer, Berlin, 2 edition, 2016. doi: 10.1007/978-3-540-30301-5.

[14] S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410, 1997. doi: 10.1109/70.585902.

[15] N. Mansard, O. Khatib, and A. Kheddar. A unified approach to integrate unilateral constraints in the stack of tasks. *IEEE Transactions on Robotics*, 25(3):670–685, 2009. doi: 10.1109/TRO.2009.2020345.

[16] F. Flacco and A. De Luca. A reverse priority approach to multi-task control of redundant robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2421–2427, 2014. doi: 10.1109/IROS.2014.6942891.

[17] A.A. Maciejewski and C.A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The International Journal of Robotics Research*, 4(3):109–117, 1985. doi: 10.1177/027836498500400308.

[18] A.S. Deo and I.D. Walker. Minimum effort inverse kinematics for redundant manipulators. *IEEE Transactions on Robotics and Automation*, 13(5):767–775, 1997. doi: 10.1109/70.631238.

[19] G.S. Chyan and S.G. Ponnambalam. Obstacle avoidance control of redundant robots using variants of particle swarm optimization. *Robotics and Computer-Integrated Manufacturing*, 28(2):147–153, 2011. doi: 10.1016/j.rcim.2011.08.001.

[20] M. Duguleana, F. Grigore Barbuceanu, A. Teirelbar, and G. Mogan. Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning. *Robotics and Computer-Integrated Manufacturing*, 28(2):132–146, 2011. doi: 10.1016/j.rcim.2011.07.004.

[21] C. Yang, S. Amarjyoti, X. Wang, Z. Li, H. Ma, and C. Y. Su. Visual servoing control of baxter robot arms with obstacle avoidance using kinematic redundancy. In H. Liu, N. Kubota, X. Zhu, R. Dillmann, and D. Zhou, editors, *Intelligent Robotics and Applications*, pages 568–580. Springer International Publishing, Cham, 2015. doi: 10.1007/978-3-319-22879-2_52.

[22] T. Petrič, A. Gams, N. Likar, and L. Žlajpah. Obstacle avoidance with industrial robots. In G. Carbone and F. Gomez-Bravo, editors, *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, pages 113–145. Springer International Publishing, Cham, 2015. doi: 10.1007/978-3-319-14705-5_5.

[23] T. Winiarski, K. Banachowicz, and D. Seredyński. Multi-sensory feedback control in door approaching and opening. In D. Filev, J. Jabłkowski, J. Kacprzyk, M. Krawczak, I. Popchev, L. Rutkowski, V. Sgurev, E. Sotirova, P. Szynkarczyk, and S. Zadrozny, editors, *Intelligent Systems'2014*, pages 57–70, Cham, 2015. Springer International Publishing. doi: 10.1007/978-3-319-11310-4_6.

[24] M. Tanaka and F. Matsuno. Modeling and control of head raising snake robots by using kinematic redundancy. *Journal of Intelligent & Robotic Systems*, 75(1):53–69, 2013. doi: 10.1007/s10846-013-9866-y.

[25] C. Ye, S. Ma, B. Li, and Y. Wang. Head-raising motion of snake-like robots. In *2004 IEEE International Conference on Robotics and Biomimetics*, pages 595–600, 2004. doi: 10.1109/ROBIO.2004.1521847.

[26] J.-A. Claret, G. Venture, and L. Basañez. Exploiting the robot kinematic redundancy for emotion conveyance to humans as a lower priority task. *International Journal of Social Robotics*, 9(2):277–292, 2017. doi: 10.1007/s12369-016-0387-2.

[27] K. Mikołajczyk, M. Szumowski, P. Płoński, and P. Żakieta. Solving inverse kinematics of humanoid robot using a redundant tree-shaped manipulator model. In *Proceedings of the 2020 4th International Conference on Vision, Image and Signal Processing*, ICVISP 2020, New York, NY, USA, 2020. Association for Computing Machinery. doi: 10.1145/3448823.3448885.

[28] M. Szumowski, M.S. Żurawska, and T. Zielińska. Preview control applied for humanoid robot motion generation. *Archives of Control Sciences*, 29(1):111–132, 2019. doi: 10.24425/acs.2019.127526.

[29] M. Szumowski, M.S. Żurawska, and T. Zielińska. Simplified method for humanoid robot gait generation. In T. Uhl, editor, *Advances in Mechanism and Machine Science*, pages 2269–2278. Springer International Publishing, 2019. doi: 10.1007/978-3-030-20131-9_224.

[30] T. Zielinska, L. Zimin, M. Szumowski, and W. Ge. Motion planning for a humanoid robot with task dependent constraints. In T. Uhl, editor, *Advances in Mechanism and Machine Science*, pages 1681–1690. Springer International Publishing, Cham, 2019. doi: 10.1007/978-3-030-20131-9_166.

[31] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pages 1620–1626. IEEE, 2003. doi: 10.1109/ROBOT.2003.1241826.

[32] S.R. Buss and J.-S. Kim. Selectively damped least squares for inverse kinematics. *Journal of Graphics Tools*, 10(3):37–49, 2005. doi: 10.1080/2151237X.2005.10129202.

[33] A. Ben-Israel and T.N.E. Greville. *Generalized Inverses, Theory and Applications*. Springer-Verlag New York, 2 edition, 2003. doi: 10.1007/b97366.

[34] P. Falco and C. Natale. On the stability of closed-loop inverse kinematics algorithms for redundant robots. *IEEE Transactions on Robotics*, 27(4):780–784, 2011. doi: 10.1109/TRO.2011.2135210.

[35] A. Colomé and C. Torras. Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements. *IEEE/ASME Transactions on Mechatronics*, 20(2):944–955, 2015. doi: 10.1109/TMECH.2014.2326304.

[36] D.N. Nenchev. Redundancy resolution through local optimization: A review. *Journal of Robotic Systems*, 6(6):769–798, 1989. doi: 10.1002/rob.4620060607.

[37] H. Zghal, R.V. Dubey, and J.A. Euler. Efficient gradient projection optimization for manipulators with multiple degrees of redundancy. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 1006–1011, vol. 2, 1990. doi: 10.1109/ROBOT.1990.126123.

[38] A. Liégeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12):868–871, 1977. doi: 10.1109/TSMC.1977.4309644.

[39] D.-S. Bae and E. Haug. A recursive formulation for constrained mechanical system dynamics: Part I. Open loop systems. *Mechanics of Structures and Machines*, 15(3):359–382, 1987. doi: 10.1080/08905458708905124.

[40] G. Rodriguez, A. Jain, and K. Kreutz-Delgado. A spatial operator algebra for manipulator modeling and control. *The International Journal of Robotics Research*, 10(4):371–381, 1991. doi: 10.1177/027836499101000406.

[41] K. Yamane and L. Nakamura. O(N) forward dynamics computation of open kinematic chains based on the principle of virtual work. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, volume 3, pages 2824–2831, 2001. doi: 10.1109/ROBOT.2001.933050.

[42] Ł. Woliński and P. Malczyk. Dynamic modeling and analysis of a lightweight robotic manipulator in joint space. *Archive of Mechanical Engineering*, 62(2):279–302, 2015. doi: 10.1515/meceng-2015-0016.

[43] M.W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. Wiley, 2005.

[44] B. Espiau and R. Boulic. On the Computation and Control of the Mass Center of Articulated Chains. Technical Report RR-3479, INRIA, August 1998.

[45] D.A. Winter. *Biomechanics and Motor Control of Human Movement*. John Wiley & Sons, Inc., 2009. doi: 10.1002/9780470549148.