Constraint-Aware Closed Loop Inverse Kinematic Control of Humanoid Upper Body Robotic System

Mutyala Gowtham Kumar
Research and Development Establishment (Engineers)
Defence Research and Development Organization
Pune, India
gowthammutyala.rde@gov.in

Nelli Anantha Surya Vamshi
Research and Development Establishment (Engineers)
Defence Research and Development Organization
Pune, India
surya13vamshi@gmail.com

Dr.Kiran Akella
Research and Development Establishment (Engineers)
Defence Research and Development Organization
Pune, India
kiranakella.rde@gov.in

doi: Doi: https://doi.org/10.37285/bsp.SACAD2025.49

Abstract—Redundant arms of humanoids helps them to perform more human like tasks. Inverse kinematics (IK) of redundant arms is challenging since there are infinite solutions for a given position and orientation (pose) of end effector. In this work, we solved IK of redundant arm using closed loop algorithm for a given pose at velocity level for elliptical trajectories. We developed closed loop algorithms based on pseudoinverse and damped least squares (DLS) methods and the constraint-awareness is introduced at velocity level using null space optimization with manipulability and joint limit avoidance as a potential function. Algorithm using DLS handled singularities well but compromising on tracking ability and that of pseudoinverse performed better in tracking test but less robust near singularities. Overall, the framework enables stable and Constraint-aware IK solutions that avoid singularities and respect joint limits at the kinematic level. The results are verified through simulations and experiments on humanoid upper body test setup.

Index Terms—7 DOF, twist, pseudoinverse, damped least squares, manipulability, joint limit avoidance

I. INTRODUCTION

Position and orientation of a rigid body in space is completely described by six degrees of freedom (DOF) [1]. A humanoid arm is redundant when it has more DOF (n) than DOF of a rigid body (m = 6). Redundancy allows the robotic arm to perform human like tasks. It also helps the arm to achieve secondary goals such as obstacle avoidance, singularity avoidance, joint limit avoidance etc. in addition to primary task of trajectory tracking. Robotic tasks are specified in task space and executed in joint space using Inverse Kinematics (IK). For redundant arms, IK is not straightforward as it has infinite solutions. Choosing one solution requires redundancy resolution. Redundancy of the system can be resolved by projecting selected optimization function on to the non-empty null space [2]. This optimization criteria can be of constraint based [3] such as joint limit avoidance, velocity limit avoidance, singularity avoidance or goal based [4] such as operation speed, load carrying capacity etc. In this approach, solution to the IK problem is split into two parts - particular solution and homogeneous solution. The particular solution of the IK problem can be found by pseudoinverse or others like DLS [5]. Corresponding homogeneous solution is obtained by projecting secondary criteria on to the null space of Jacobian matrix. Both pseudoinverse and singularity robust inverse (SRI) [6] using DLS are orthogonal to the null space of Jacobian matrix. Therefore, null space solution does not affect the main task solution. This can be easily visualized by keeping your arm on the table and moving your elbow. Here the main task solution is keeping end effector (palm) velocity zero and the null space solution is your elbow motion, also called self-motion. While solving for velocity inverse kinematics, velocity errors accumulate making end effector deviate from original position, resulting in position tracking errors [1]. Hence, a feedback term is necessary. As an outcome of closing the position loop using feedback, this approach is called closed loop inverse kinematics (CLIK) algorithm [7], [8]. CLIK can be at velocity or acceleration levels. In Wang et al. [9], IK is solved using CLIK for position tracking without considering orientation and joint limit avoidance is used to resolve redundancy. In some popular open source planners such as MoveIt, often KDL (kinematics and dynamics library) [10] is used to solve IK using damped least squares. Typical issue of such approaches is failure of IK calculation. This can be avoided by adding null space optimization term for any redundant robot instead of using default KDL [11]. New improvements are being made to MoveIt library which primarily focuses on redundancy resolution, where robots with additional degrees of freedom can use different configurations to perform same task. Also solvers like Bio-IK in MoveIt facilitates user to specify cost functions that can prioritize certain solutions to IK [12]. In this paper, we have solved IK and kinematically controlled the arm using closed loop algorithm. This type of position controller, without a dynamic model or acceleration term is used by DARPA robotics challenge winner HUBO+ [13]. The authors assumed that in most cases, the arm does not move rapidly. They experimentally proved position controller with open loop force control is sufficient to perform specific tasks of the challenge. In this paper, we have generated trajectories for pose and twist for an elliptical trajectory and solved main task solution using pseudoinverse and DLS methods. Constraint-awareness is introduced by using manipulability or joint limit avoidance as optimization function for secondary criteria. Using DLS method, appropriate position gains and joint limit avoidance as optimization criteria we could track a particular trajectory avoiding singular configurations and respecting joint limits. In this paper, section 2 describes the kinematic model of humanoid upper body. Section 3 describes trajectory generation. In section 4, CLIK algorithm and the control block are described. Section 5 presents computer simulations. Hardware implementation and discussions are in section 6 and section 7 respectively.

Table 1: DH parameters for right and left arms

Link	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	$\pi/2$	l_1	$\theta_1+\pi/2$
2	0	$\pi/2$	0	$\theta_2-\pi/2$
3	0	$\pi/2$	l_2	$\theta_3+\pi/2$
4	0	$-\pi/2$	0	$\theta_4-\pi/2$
5	0	$-\pi/2$	l_3	θ_5
6	0	$\pi/2$	0	θ_6
7	0	$-\pi/2$	l_4	θ_7

Link	a_{i-1}	α_{i-1}	d_i	θ_i
8	0	$\pi/2$	$-l_1$	$\theta_8+\pi/2$
9	0	$\pi/2$	0	$\theta_9-\pi/2$
10	0	$\pi/2$	l_2	$\theta_{10}+\pi/2$
11	0	$-\pi/2$	0	$\theta_{11}-\pi/2$
12	0	$-\pi/2$	l_3	θ_{12}
13	0	$\pi/2$	0	θ_{13}
14	0	$-\pi/2$	l_4	θ_{14}

II. KINEMATIC MODEL OF HUMANOID UPPER BODY

A typical humanoid robot upper body consisting of two redundant 7-DOF arms with all joints rotational is chosen for this study. We used DH frames to represent the system with zeroth frame as base frame. DH parameters obtained by using modified DH convention [14], [15] are shown in Table I. CAD image of initial configuration where elbow is at 90 degrees and rest of joints at zero degrees is illustrated in Fig. 1.

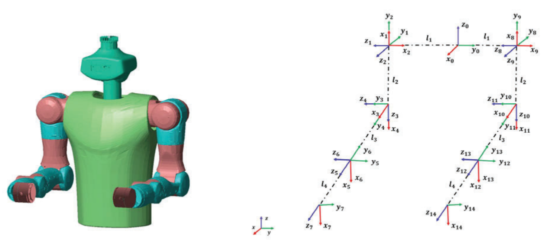


Fig. 1: Configuration diagram of humanoid upper body

III. TRAJECTORY GENERATION

Trajectory generation requires initial rotational matrix (R_i), final rotational matrix (R_f), initial position (P_i), final position (P_f) and time scaling parameter (s) as inputs. In our work, s is taken as time, t. Instantaneous rotational matrix is calculated from the expression $R = R_i exp(log(R_i^T R_f)s$. Corresponding Euler angles (α, β, γ) are calculated using rotation matrix to Euler angle converter block (Rotm2eul). The angular velocities ($\omega_x, \omega_y, \omega_z$) are extracted from skew symmetric matrix $\dot{R}R^T$[1]. Position trajectory (x, y, z) is calculated using Cartesian trajectory block. Linear velocities $(\dot{x}, \dot{y}, \dot{z})$ are calculated by differentiating corresponding positions. In this way, we were able to generate pose, $X_d = [x, y, x, \alpha, \beta, \gamma]^T$ and twist, $V_d = [\dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$ as a function of time. Note that $\dot{X}_d \neq \mathrm{v_d}$. Trajectory generation is illustrated in Fig. 2.

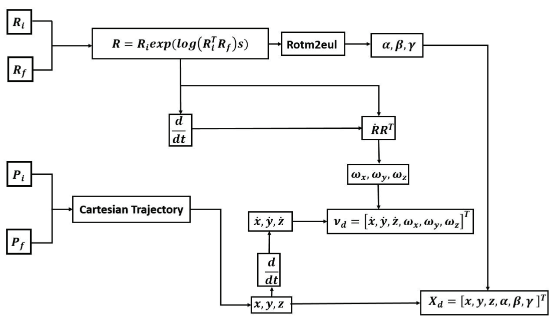


Fig. 2: Trajectory generation

IV. INVERSE KINEMATICS

A. Inverse Kinematics Formulation

Inverse kinematics involves finding required joint positions (θ) and joint velocities $(\dot{\theta})$ for a given end effector pose (X)

and twist (V). Jacobian Matrix, J maps joint velocity space to end effector space. Velocity kinematics can therefore be expressed as:

$$V = J(\theta)\dot{\theta} \tag{1}$$

In our case, each arm has seven degrees of freedom. Due to non-square nature of jacobian (6×7), IK problem has to be solved differently than the standard matrix inverse used for square matrices. Two approaches explored here are:

$$\dot{\theta} = J^{+}(\theta)V \tag{2}$$

and

$$\dot{\theta} = J^{*}(\theta)V \tag{3}$$

where $J^{+} = J^{T}(JJ^{T})^{-1}$ is Moorse Penrose pseudoinverse and $J^{*} = (J^{T}J + \lambda^{2}I)^{-1}J^{T}$ is called singularity robust inverse (SRI) based on DLS.

Equation (1) has infinitely many solutions since it has more unknowns ($n = 7$) than number of equations ($m = 6$) as the system is redundant. For such systems, non-empty null space always exists. In our case, null space refers to set of joint velocities that yield zero task space velocities at current robot configuration. Note that this null space solution does not affect the main task solution since column space of J, denoted by $R(J)$ and Null space of J, denoted by $N(J)$ are orthogonal complements. This means all the solutions which are orthogonal to $R(J)$ are in $N(J)$. Hence, pseudoinverse (J^{+}) is orthogonal to $N(J)$. It has been proved that even SRI (J^{*}) using DLS is orthogonal to $N(J)$ [6]. With this background, final solution for inverse kinematics can be formulated as a sum of main task solution (or particular solution) and a homogeneous solution [2]. Main task solution can be solved using pseudoinverse or DLS and homogeneous solution can be found by projecting certain optimization criteria on to the null space of J. Hence the final solution for a redundant system can be written by pseudoinverse (4) and DLS (5) respectively as:

$$\dot{\theta} = J^{+}(\theta)V + N(J)\dot{\theta}_0 \tag{4}$$

and

$$\dot{\theta} = J^{*}(\theta)V + N(J)\dot{\theta}_0 \tag{5}$$

where $N(J)$ is null space projection given by $(I - J^{+}(\theta)J(\theta))$ and $\dot{\theta}_0$ is an arbitrary vector. Solving inverse kinematics using (4) or (5) leads to accumulating velocity errors leading to position tracking errors. So pose feedback is used for tracking correction. This approach is called closed loop inverse kinematics (CLIK) algorithm. Thus, (4) and (5) after including pose feedback can be written as (6) and (7):

$$\dot{\theta} = J^{+}(\theta)(\nu_d + K_p(X_d - X)) + N(J)\dot{\theta}_0 \tag{6}$$

or

$$\dot{\theta} = J^{*}(\theta)(\nu_d + K_p(X_d - X)) + N(J)\dot{\theta}_0 \tag{7}$$

where (X_d, ν_d) are desired end effector pose and twist obtained from trajectory block, X is actual end effector pose and K_p is position gain matrix. Appropriate choice of K_p makes sure that error converges to zero. Error dynamics can be found in [9]. The control block diagram is presented in Fig.3.

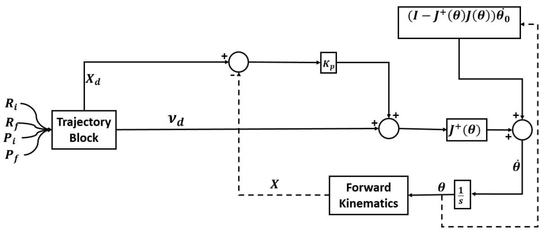


Fig. 3: Kinematic control of humanoid arm

B. Redundancy Resolution

Joint velocities in null space does not contribute to end effector velocities since they are solutions of homogeneous equations $J\dot{\theta}_0 = 0$. These motions are null space motions or self-motions. With the help of these self-motions, it is possible to perform secondary tasks in addition to main task. Most common subtasks are joint limit avoidance, singularity avoidance, obstacle avoidance etc. In this work, we used singularity avoidance and joint limit avoidance as additional tasks and the redundancy is resolved based on gradient projection method [16].

1) Singularity Avoidance: When the arm is approaching singular configurations, very large joint velocities are required for very small task space movements. At exact singular configurations infinite joint velocities are required. In other way, at singular configurations the robot arm loses its ability to move in particular direction. It is possible to mathematically measure how close the given configuration is to singularity. The manipulability ellipsoid helps us to determine the directions of end effector in which least and largest effort is required. The volume of the ellipsoid at a particular configuration can give us the non-directional measure of distance from the singularity. Larger the ellipsoid volume, farther the configuration is from singularity. This volume is proportional to $\sqrt{\det(JJ^{T})}$ [1]. By using singular value decomposition (SVD) of J, we can obtain the manipulability measure, $V(\theta)$: (8)

$$V(\theta) = \sqrt{\det(JJ^{T})} = \sigma_1\sigma_2\sigma_3\sigma_4\sigma_5\sigma_6 \tag{8}$$

ISBN: 978-93-47311-06-2

where $\sigma_1 \dots \sigma_6$ are diagonal elements of D matrix in SVD of $J = UDV^T$. The manipulability measure decreases when the configuration is approaching singular configuration and becomes zero at singular configurations. Maximization of this measure by gradient projection method helps to avoid singularities which can be avoidable.

$$\dot{\theta}_0 = \beta \left(\frac{\partial V(\theta)}{\partial(\theta)} \right) \tag{9}$$

where $\beta > 0$ determines the speed of optimization.

2) Joint Limit Avoidance: The objective function, $W(\theta)$ for joint limit avoidance is the distance of present configuration from the mechanical joint limits and it is minimized by using gradient projection method (11).

$$W(\theta) = \frac{(\theta - \theta_{mean})^2}{(\theta_{max} - \theta_{min})^2} \tag{10}$$

$$\dot{\theta}_0 = \beta \left(\frac{\partial W(\theta)}{\partial(\theta)} \right) \tag{11}$$

where $\theta_{max}, \theta_{min}$ are maximum, minimum allowable joint angles and $\theta_{mean} = (\theta_{max} + \theta_{min})/2$.

V. SIMULATIONS

Computer simulations are performed using MATLAB [17] on the right arm of humanoid upper body. Elliptical trajectory with minimal change in orientation is taken as desired trajectory. Parameters used for trajectories are shown in Table II. Ellipse trajectory chosen is in $Y - Z$ plane at $x = 0.386$. Center (h, k) is assumed to be at $(0.405, -0.244)$, major and minor axes (a, b) at $(0.08, 0.05)$. Simulations are presented in two sets. Set 1 compares performance of pseudoinverse and DLS methods for ellipse trajectory with manipulability as optimization criteria. Set 2 illustrates performance of DLS method in tracking ellipse with and without joint limit avoidance as optimization criteria. Tracking performance with change in K_p is also studied in set 2. The value of K_p is obtained by trial and error.

Table 2: Parameters for ellipse trajectory

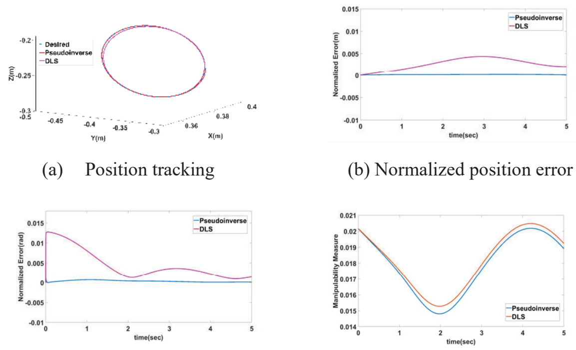
Parameter	Ellipse
R_i	$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}$
R_f	$\begin{pmatrix} 0 & -0.174 & 0.985 \\ 0 & 0.985 & 0.174 \\ -1 & 0 & 0 \end{pmatrix}$
P_i	$\begin{pmatrix} 0.386 \\ -0.325 \\ -0.244 \end{pmatrix}$
P_f	$\begin{pmatrix} 0.386 \\ -0.325 \\ -0.244 \end{pmatrix}$
Cartesian trajectory	$(y = h + a\cos 2\pi s), (z = k + b\sin 2\pi s)$
Rotation trajectory	$R = R_i \exp(\log(R_i^T R_f)s)$
K_{p1}	$[256, 256, 256, 2.56, 7.11, 64] * 10^{-4}$
K_{p2}	$[256, 256, 256, 2.56, 7.11, 64] * 10^{-2}$

A. Simulation set 1

In first simulation, main task solution with diagonal values of $K_p = K_{p1}$ is calculated based on pseudoinverse and DLS $(\lambda^2 = 0.001)$ methods. Null space solution is calculated by maximizing manipulability measure for main task solution. In this way, out of many solutions available for the redundant system, trajectory is tracked (main solution) in such a way that configuration is farthest from singularity (null space solution). Tracking ability and manipulability measure along desired trajectory (for pseudoinverse and DLS) of an ellipse is shown in Fig.4. Normalized position error is $\sqrt{\frac{(x_d - x)^2 + (y_d - y)^2 + (z_d - z)^2}{3}}$ and normalized Euler angle error is $\sqrt{\frac{(\alpha_d - \alpha)^2 + (\beta_d - \beta)^2 + (\gamma_d - \gamma)^2}{3}}$. Where (x_d, y_d, z_d) and (x, y, z) are desired and actual positions. $(\alpha_d, \beta_d, \gamma_d)$ and (α, β, γ) are desired and actual Euler angles.

B. Simulation set 2

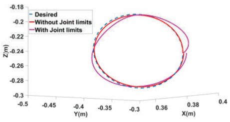
Since pseudoinverse fails near singularities, we used manipulability measure as optimization criteria in simulation set 1. On the contrary, DLS method is robust against singularities. However, it requires a non-physical parameter $'\lambda'$ and also has lower trajectory tracking ability. In this work, using feedback correction, we have circumvented this issue and improved the tracking ability of DLS by carefully choosing λ and K_p. This gave us the additional freedom to use another optimization criteria. We can then perform two simultaneous tasks i.e. trajectory tracking and joint limit avoidance. Using DLS



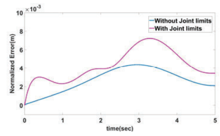
(a) Position tracking (b) Normalized position error

(c) Normalized Euler angle (d) Manipulability

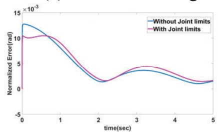
Fig. 4: Ellipse trajectory tracking using pseudoinverse/DLS for main task solution and manipulability optimization for null space solution



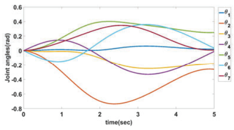
(a) Position tracking

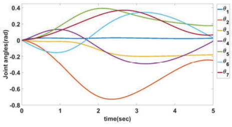


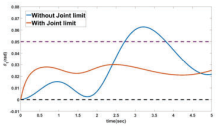
(b) Normalized position error



(c) Normalized Euler angle error


(d) Joint angles without joint limit optimization


(e) Joint angles with joint limit optimization



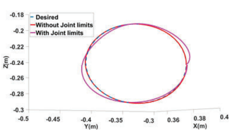
(f) θ_1 without and with joint limit

Fig. 5: Ellipse trajectory tracking using DLS with K_{p1} for main task solution and joint limit avoidance for null space solution

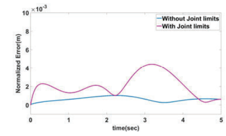
($\lambda^2 = 0.001$), elliptical trajectory tracking with $K_p = K_{p1}$ (with and without joint limits) and K_{p2} (with and without joint limits) is studied. (see Fig. 5 and Fig. 6).

VI. HARDWARE IMPLEMENTATION

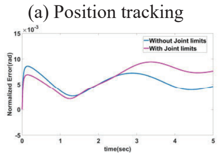
Proposed control strategy was implemented on dual-arm robotic platform, with each arm consisting of 7 degrees of freedom (see Fig 7). We used cascaded position control at lower level [18], which regulates joint trajectories. On top of this inner loop, an outer-loop CLIK controller was developed to generate reference joint positions based on task space errors. The control framework was developed in Python with ROS2 serving as the



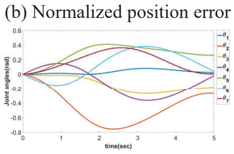
(a) Position tracking

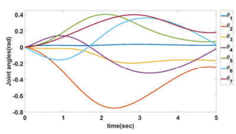


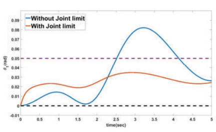
(b) Normalized position error



(c) Normalized Euler angle error


(d) Joint angles without joint limit optimization


(e) Joint angles with joint limit optimization



(f) θ_1 without and with joint limit

Fig. 6: Ellipse tracking using DLS with K_{p2} for main task solution and joint limit avoidance for null space solution

communication middleware for real-time interaction between the controller and the robotic system. Experiments were executed at a control frequency of 100Hz, ensuring smooth task execution. Implementation modules are as follows:

1) Controller Module: This module contained mathematical formulation of outer-loop controller, including DH parameter based forward kinematics, Jacobian computation, and twist generation. Computed joint values were published as reference inputs to inner position loop of the hardware.
2) Motion Initialization Module: This module is used to define initial transformation matrices, final transformation matrices, and motion duration parameters. By separating defined motion from the control dynamics, framework provides flexibility in executing different dual arm tasks. The combined inner-outer loop design allowed stable operation of the dual arm system, where the hardware's cascaded joint controllers ensured low-level accuracy, and the CLIK outer loop achieved reliable end-effector trajectory tracking.

VII. DISCUSSIONS

In simulation set 1, for ellipse trajectory (see Fig. 4(a)), tracking ability of pseudoinverse method is better than DLS method. It is due to parameter λ. Both normalized position and Euler angle errors are more for DLS method. Another interesting thing to note is how manipulability changes in case of ellipse trajectory (see Fig. 4(d)). Manipulability decreased initially as arm began to stretch and then it increased when arm began to come closer to initial configuration. We observed

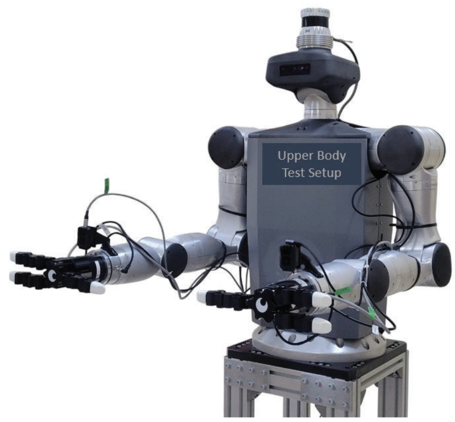


Fig. 7: Humanoid upper-body test setup

that manipulability measure at any position is more for DLS method than pseudoinverse method. In simulation set 1, while using pseudoinverse for main task solution, we have to use manipulability as optimization criteria as J^+ becomes ill conditioned near singularities. In simulation set 2, use of SRI based on DLS for main task solution takes care of singularity avoidance while compromising on tracking. Therefore, we used pose feedback with suitable value of K_p to improve tracking performance. It also allowed us to use another optimization criteria other than singularity avoidance. The tracking ability for ellipse trajectory (main task solution) using DLS ($\lambda^2 = 0.001$) and joint limit avoidance (secondary solution) with different K_p can be seen in Fig. 5 and Fig. 6. To check joint limit avoidance, minimum and maximum values of θ_1 are restricted to $[0,0.05]$ radians in secondary criteria. It can be seen in Fig. 5(f) and Fig. 6(f) that joint limit has not been exceeded. In this way, we can achieve better tracking, avoid singularities and also be within joint limits.

VIII. CONCLUSIONS

In this work, we have solved inverse kinematics for redundant humanoid arm using closed loop algorithm. Trajectories are generated for ellipse using trajectory block, which gives pose and twist of end effector with respect to time. We used pseudoinverse and DLS methods for main task solution and manipulability measure or joint limit avoidance as optimization criteria for null space solution. Simulations showed that pseudoinverse method performed better in tracking tests while DLS method has advantage of manipulability. Tracking ability of DLS method was improved by changing position gain. With suitable position gain, damping parameter and joint limit avoidance as optimization criteria, we could achieve singularity avoidance, minimal tracking errors and additional criteria like joint limit avoidance at the kinematic level itself.

IX. FUTURE WORK

One important problem in Robotics is when we feed joint angles to controllers, often joint velocity limits exceed, resulting in jerky motions. These joint velocity limits can also be avoided by using null space optimization of certain potential functions. These functions can be explored. Furthermore, this method can be extended to manipulation cum locomotion i.e. using extra degrees of freedom provided by legs of humanoid (see Fig. 8) to achieve the primary and secondary tasks.

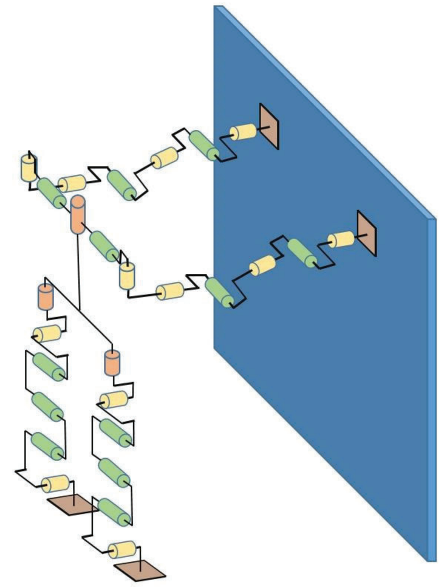


Fig. 8: Humanoid configuration

REFERENCES

[1] Kevin M. Lynch and Frank C. Park. *Modern robotics*. Cambridge University Press, 2017.

[2] Gilbert Strang. *Introduction to linear algebra*. SIAM, 2022.

[3] Chetan Kapoor, Murat Cetin, and Delbert Tesar. Performance based redundancy resolution with multiple criteria. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 80319, page V01BT01A018. American Society of Mechanical Engineers, 1998.

[4] Richard Hooper and Delbert Tesar. Motion coordination based on multiple performance criteria with a hyper-redundant serial robot example. In *Proceedings of Tenth International Symposium on Intelligent Control*, pages 133–138. IEEE, 1995.

[5] Samuel R Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation*, 17(1-19):16, 2004.

[6] Arati S Deo and Ian D Walker. Robot subtask performance with singularity robustness using optimal damped least-squares. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 434–435. IEEE Computer Society, 1992.

[7] Bruno Siciliano. A closed-loop inverse kinematic scheme for on-line joint-based robot control. *Robotica*, 8(3):231–243, 1990.

[8] Pietro Falco and Ciro Natale. On the stability of closed-loop inverse kinematics algorithms for redundant robots. *IEEE Transactions on Robotics*, 27(4):780–784, 2011.

[9] Jingguo Wang, Yangmin Li, and Xinhua Zhao. Inverse kinematics and control of a 7-dof redundant manipulator based on the closed-loop algorithm. *International Journal of Advanced Robotic Systems*, 7(4):37, 2010.

[10] Sachn Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE robotics & automation magzine*, 19(1):18–19, 2012.

[11] Karan Khokar, Patrick Beeson, and Rob Burridge. Implementation of kdl inverse kinematics routine on the atlas humanoid robot. *Procedia Computer Science*, 46:1441–1448, 2015.

[12] Philipp Ruppel, Norman Hendrich, Sebastian Starke, and Jianwei Zhang. Cost functions to specify full-body motion and multi-goal manipulation tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3152–3159. IEEE, 2018.

[13] Jeongsoo Lim, Inho Lee, Inwook Shim, Hyobin Jung, Hyun Min Joe, Hyoin Bae, Okkee Sim, Jaesung Oh, Taejin Jung, Seunghak Shin, et al. Robot system of drc-hubo+ and control strategy of team kaist in darpa robotics challenge finals. *Journal of Field Robotics*, 34(4):802–829, 2017.

[14] Jacques Denavit and Richard S Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. 1955.

[15] John J Craig. *Introduction to robotics: mechanics and control, 3/E*. Pearson Education India, 2009.

[16] Hajer Zghal, Rajiv V Dubey, and James 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. IEEE, 1990.

[17] The MathWorks Inc. Matlab version : 9.13.0 (r2022b), 2022.

[18] *https://www.synapticon.com/en/support/dokumentation*. 2025.