2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)
Banff, Alberta, Canada, July 12–15, 2016

Development of a Quadruped Robot with Redundant DOFs for High-degree of Functionality and Adaptation

Bokeon Kwak, Hyunkyoo Park, and Joonbum Bae, *Member, IEEE*

***Abstract*— This paper presents a quadruped robot with redundant degrees of freedom (DOFs), which exploits its kinematic redundancy for various types of locomotion and manipulation. Unlike previously developed quadruped robots, the proposed robot can change its body posture and suitably adapt to different environments. For example, the robot can walk on a plain terrain, pass through a narrow gap, surmount an obstacle, perform a simple task by using one of its legs as a manipulator, etc. Inverse kinematics of each leg was solved by using a newly proposed method: Improved Jacobian Pseudoinverse (IJP) algorithm. Also, a static stability of the robot was dealt with using a combined Jacobian of a center of mass (COM) and centroid of a support polygon. Performance of the proposed robot was verified in both simulations and experiments.**

I. Introduction

A kinematically redundant manipulator is typically exploited to perform multiple tasks such as avoiding the joint limit, obstacles in the workspace, singularities, etc [1]. Many literatures have proposed several algorithmic approaches and solutions to control a redundant manipulator. For example, in [2] a new scheme using the weighted least-norm (WLN) solution was proposed to guarantee joint limits avoidance. Compared with the gradient projection method (GPM), which projects a gradient of performance criterion on the null space of the Jacobian, WLN scheme could avoid unnecessary self motions and oscillation when it was implemented in the 7 DOFs redundant manipulator. Solving inverse kinematic problems for redundant manipulators under the two constraints (avoidance of obstacle and joint limits) was introduced in [3] using the extended Jacobian matrix. Also, a general scheme to manage multiple tasks using a task priority strategy for redundant robots was studied in [4]. Widely used closed-loop inverse kinematics methods are well summarized in [5] including damping and filtering of singular values with two enhancements. A rather different approach was studied in [6] using a modified Newton-Raphson method with a step size control technique.

When controlling a manipulator, it is also important to consider a compensation of velocity and acceleration saturation of a joint. In [7] a method of joint velocity/acceleration redistribution for a redundant manipulator was introduced to preserve the joint velocity/acceleration within their limits. This was achieved by including compensation joint velocities and accelerations in the null space of the generalized inverse of the Jacobian matrix. A similar, but, computationally more efficient approach called Saturation in the Null Space (SNS) iterative algorithm was proposed in [8]. The SNS algorithm saturated only the most violating joint at a time, and proceeded an optimal task scaling when the given task trajectory was unfeasible.

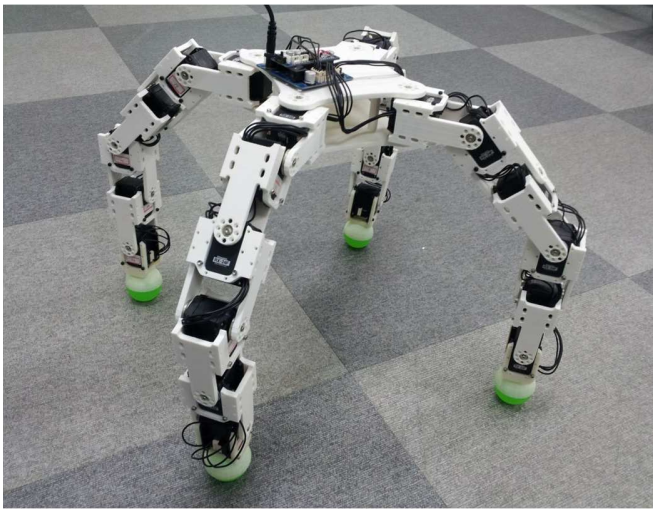


Fig. 1: Proposed quadruped robot: each leg has 6 joints, and the whole system is composed of 24 DOFs.

To take advantage of a redundant manipulator's dexterity in locomotion, a robot that has four legs with 6 joints for each leg is proposed as shown in Fig. 1. Compared with the similar size of quadruped robot like LittleDog [9], [10], our robot is developed for high-degree of functionality and adaptation in various environments. For example, our robot can walk on a plain terrain, pass trough a narrow gap by changing its body posture, and cross over an obstacle. Also the robot can perform a simple task, such as clearing a path, by using one of its leg as a manipulator. All this scenarios were tested in the simulations, and experiments were conducted to verify its feasibility.

Another similar robot named RoboSimian has highly dexterous four 7-DOF limbs for both locomotion and manipulation [11]. To control such high DOF limbs, an inverse kinematics look-up table together with gradient-based algorithm was used. Even though the proposed approach converged to a solution rapidly, it accompanied with large computational cost. Here, we mainly relied on a differential kinematics to control the robot with less computational burden including a newly proposed method: Improved Jacobian Pseudoinverse

This work was supported by the 2016 Research Fund (1.160005.01) of UNIST (Ulsan National Institute of Science and Technology).

B. Kwak, H. Park, and J. Bae (corresponding author) are with the Bio-Robotics and Control (BiRC) Laboratory, Department of Mechanical Engineering, UNIST, Ulsan, Korea e-mail: {jeff624, hkpark910, jbbae}@unist.ac.kr

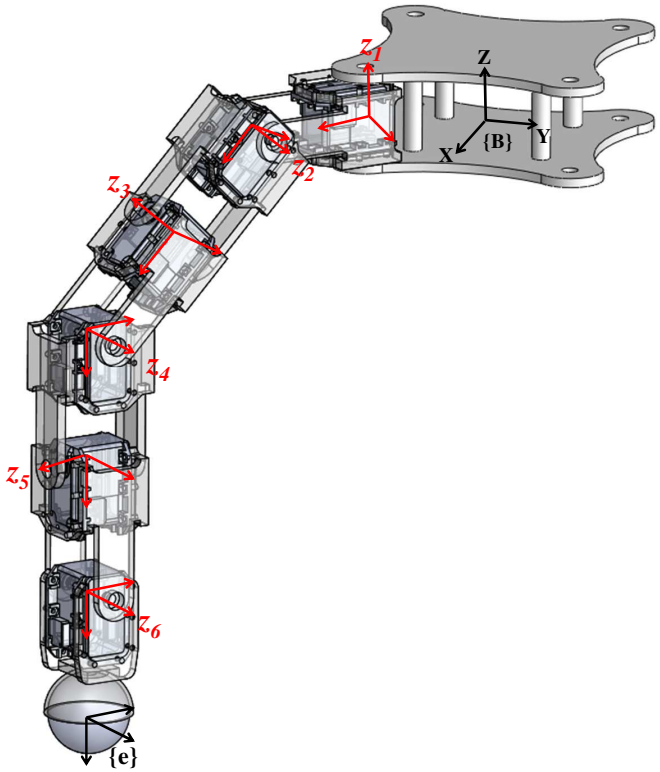


Fig. 2: Attached reference frames on the robot ({B}: the robot's body frame, {e}: end-effector frame). z-axis of the red frames coincide with the rotational axis of each joint.

(IJP) algorithm.

The remainder of this paper is organized as follows. In Section II, a IJP algorithm is proposed to solve inverse kinematics of the legs. A combined Jacobian of center of mass (COM) and centroid for a static stability is introduced in Section III. Various simulations and experiment results are given in Section IV, and V in each. Conclusions and future work are summarized in Section VI.

II. Improved Jacobian Pseudoinverse (IJP) Algorithm

As previously mentioned, each leg of the proposed robot has 6 joints as shown in Fig. 2. The kinematic relation between the robot's body frame $\{\mathbf{B}\}$ and the end-effectors frame $\{\mathbf{e}\}$ was found according to Denavit-Hartenburg (DH) convention. Note that the geometric configurations, and the attached reference frames of other legs are all identical. Currently, our robot has no sensory feedback system and all trajectories are generated in open-loop manner during the simulations and experiments. Hence, at least to ensure all the joints are always within their rotational limits, a task space vector that composed of three variables ($X-Y-Z$ positions) is given to each leg to exploit the remaining kinematic redundancy of three.

A differential kinematics is commonly used to control a manipulator since it relates joint space velocity with a task space velocity through a Jacobian [12], [13]. In this work, we used the Moore-Penrose pseudoinverse $\boldsymbol{J}^+$, which yields the lowest magnitude of joint velocities, due to its analytical tractability, continuity, and uniqueness of the solution [14].

By considering a single leg of the robot, the joint position vector $\boldsymbol{q} \in \mathbb{R}^6$ at t_{k+1} can be obtained using the Euler integration as $\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \dot{\boldsymbol{q}}(t_k)\Delta t$, where $\dot{\boldsymbol{q}} \in \mathbb{R}^6$ is joint velocity vector and Δt is a time step. From [13], the joint velocity of a redundant manipulator is calculated as follows:

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{J}^+(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e}) + (\boldsymbol{I}_6 - \boldsymbol{J}^+\boldsymbol{J})\dot{\boldsymbol{z}}_0 \\ &\text{where } \boldsymbol{J}^+ = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T + \alpha^2\boldsymbol{I}_3) \\ &\text{, and } \dot{\boldsymbol{z}}_0 = -\beta\Big(\frac{q_i - \bar{q}_i}{q_{i,\max} - q_{i,\min}}\Big)^T \end{aligned} \tag{1}$$

From (1), $\boldsymbol{J}^+ \in \mathbb{R}^{6\times3}$ is a right pseudoinverse of the Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{3\times6}$, $\dot{\boldsymbol{x}}_d \in \mathbb{R}^3$ is desired velocity vector in the task space seen from the body frame $\{\mathbf{B}\}$, $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ is a positive definite diagonal matrix, $\boldsymbol{e} \in \mathbb{R}^3$ is an error between the end-effector's desired position and the current position, $\boldsymbol{I}$ is an identity matrix, and $\dot{\boldsymbol{z}}_0 \in \mathbb{R}^6$ is a gradient projector on the null space of $\boldsymbol{J}$. To prevent the Jacobian $\boldsymbol{J}$ from losing its rank, a damped least square [15] was used where α is a damping factor. Also, to exploit the redundancy of the leg, $\dot{\boldsymbol{z}}_0$ is defined as (1), so the joints are forced to stay within their limits. β is a constant coefficient, q_i is a current position of joint i, $\bar{q}_i$ is its center position, and $q_{i,\max}$ and $q_{i,\min}$ are its maximum and minimum position respectively.

One can measure the distance from a singularity of a (redundant) manipulator at current posture using a condition number (CN) of the Jacobian, $\kappa(\boldsymbol{J}^+)$, as below.

$$\kappa(\boldsymbol{J}^+) = \frac{\sigma_{\max}(\boldsymbol{J}^+)}{\sigma_{\min}(\boldsymbol{J}^+)} \tag{2}$$

In (2), $\sigma_{\max}$ and $\sigma_{\min}$ are the maximum and minimum singular values of the Jacobian $\boldsymbol{J}^+$, obtained from singular value decomposition respectively. Using a Jacobian Pseudoinverse (JP) algorithm of (1) can generate a desired trajectory accurately as long as the condition number is small. However, it shows a large error when the condition number is large. For example, Fig. 3-(a) shows the calculated trajectory of the robot's leg using (1) when its condition number at initial posture is 6.03, and the corresponding error is almost zero as shown in Fig. 3-(b). On the other hand, condition number at the initial posture in Fig. 3-(c) is 10156, and incurs a large error in the early stage of the trajectory as shown in Fig. 3-(d).

To reduce the sharp error peak, we proposed a Improved Jacobian Pseudoinverse (IJP) algorithm. The key idea is calculating again the given trajectory in reverse order after finishing the calculation in normal direction. The detailed steps are described as follows:

STEP0 : check whether the CN of the leg at initial posture is bigger than a threshold ϵ_1. If CN > ϵ_1 go to the **STEP1**, whereas, if CN < ϵ_1 calculate joint position as usual using (1) and done.

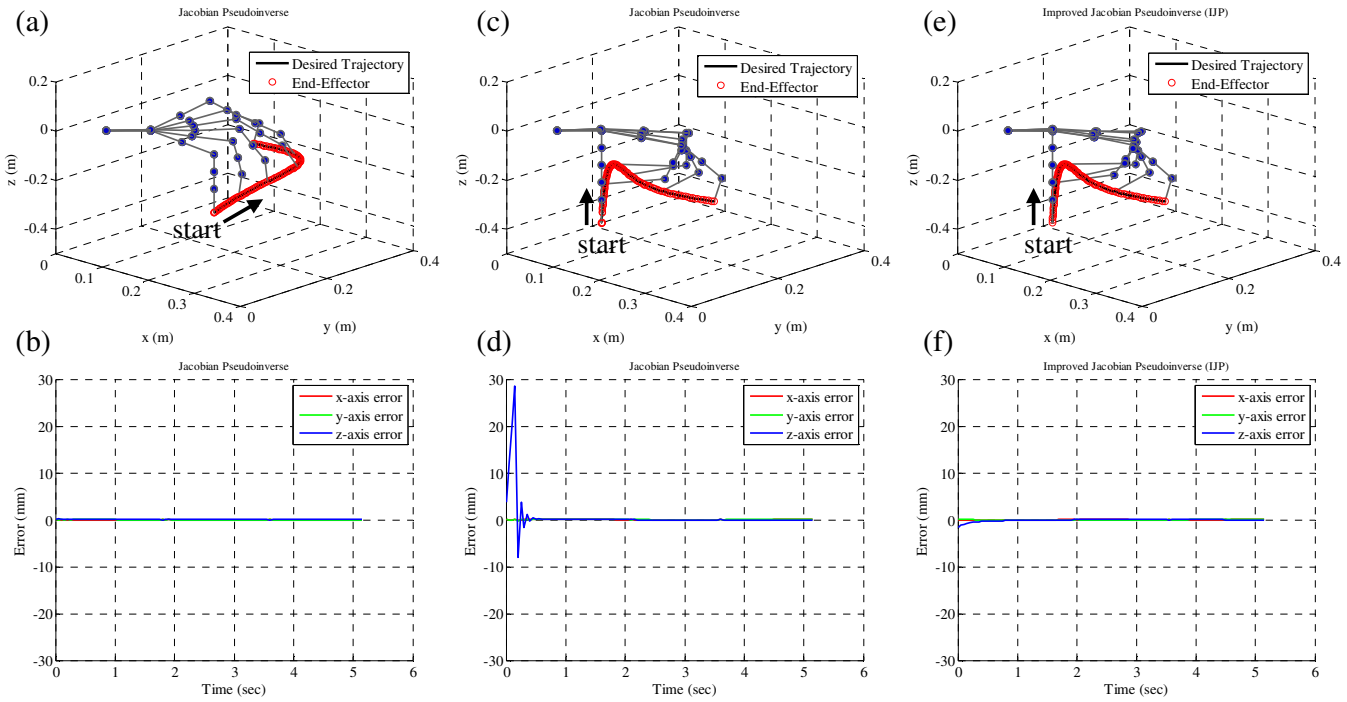


Fig. 3: Simulation results of the inverse kinematics of the robot's leg and the corresponding error with different values of condition numbers. (a) & (b): initial condition number at initial is 6.03, and JP is used, (c) & (d): initial condition number is 10156, and JP is used, (e) & (f): initial condition number is 10156, and IJP is used.

STEP1 : for a given desired trajectory $\boldsymbol{x}_d$ and a initial joint position $\boldsymbol{q}_{init}$, calculate the last joint position $\boldsymbol{q}_{end}$ at the end of $\boldsymbol{x}_d$ using (1).

STEP2 : obtain $\boldsymbol{x}_{d,rev}$ by reversing $\boldsymbol{x}_d$, and calculate the series of joint positions in reverse order $\boldsymbol{q}_{rev}$ from $\boldsymbol{x}_{d,rev}$ start from $\boldsymbol{q}_{end}$ using (1). Note that $\boldsymbol{q}_{end}$ is now the initial joint position of $\boldsymbol{x}_{d,rev}$.

STEP3 : reverse $\boldsymbol{q}_{rev}$ and obtain $\boldsymbol{q}$, which is a series of joint positions to smoothly follow $\boldsymbol{x}_d$ with a reduced error at the early stage of $\boldsymbol{x}_d$.

STEP4 : while calculating $\boldsymbol{q}_{rev}$ in **STEP2**, if CN at the beginning of $\boldsymbol{x}_{d,rev}$ is higher than a threshold ϵ_2 (equivalently, if the CN at the end of $\boldsymbol{x}_d$ is higher than ϵ_2), then calculate again only the last part of $\boldsymbol{q}$ start from a corresponding point in $\boldsymbol{x}_d$.

By using IJP, we could minimize the error as shown in Fig. 3-(f) when the leg is commanded to follow the exactly same trajectory as in Fig. 3-(c). Note that all the joints are always within their physical limits during the simulations due to $\dot{\boldsymbol{z}}_0$ in (1).

III. Combined COM and Centroid Jacobian

Since our robot has relative long legs (44 cm) compared with its body length (11.5 cm), the robot tends to easily fall down when it swings the leg. To achieve a static stability, we coincide the center of mass (COM) of the robot with a centroid of a support polygon (SP) formed by the remaining three support feet [16] whenever a leg is about to swing. A zero-moment point (ZMP) [17] is widely considered to maintain a dynamic stability [18], but this was not considered here due to the relatively slow motion of the robot. By assuming the COM of each link lies exactly its center and no foot slippage occurs, we defined a vector $\boldsymbol{P}_S \in \mathbb{R}^3$ in the robot's body frame $\{\mathbf{B}\}$ as follows:

$$\boldsymbol{P}_S = \boldsymbol{P}_{\mathrm{COM}} - \boldsymbol{P}_{\mathrm{Cent}} \tag{3}$$

where $\boldsymbol{P}_{\mathrm{COM}} \in \mathbb{R}^3$, and $\boldsymbol{P}_{\mathrm{Cent}} \in \mathbb{R}^3$ are the position vectors of the robot's COM, and the centroid of the SP viewed from the body frame. Note that $\boldsymbol{P}_{\mathrm{COM}}$ and $\boldsymbol{P}_{\mathrm{Cent}}$ in the body frame can be easily found using the forward kinematics. In similar manner as (1), a joint velocity vector of the whole body $\dot{\boldsymbol{q}}_{\mathrm{All}} \in \mathbb{R}^{24}$ can be found:

$$\dot{\boldsymbol{q}}_{\mathrm{All}} = \boldsymbol{J}_S^{+}(\dot{\boldsymbol{P}}_{S,d} + \boldsymbol{K}\boldsymbol{e}) + (\boldsymbol{I}_{24} - \boldsymbol{J}_S^{+}\boldsymbol{J}_S)\dot{\boldsymbol{z}}_0 \tag{4}$$

with obvious meanings of $\boldsymbol{J}_S \in \mathbb{R}^{3\times 24}$, $\boldsymbol{J}_S^{+} \in \mathbb{R}^{24\times 3}$, $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{e} \in \mathbb{R}^3$, and $\boldsymbol{z}_0 \in \mathbb{R}^{24}$. $\dot{\boldsymbol{P}}_{S,d} \in \mathbb{R}^3$ is a time derivative of desired $\boldsymbol{P}_S$, and by letting $\boldsymbol{P}_{S,d} = [0, 0, z_S]^T$, where z_S is an arbitrary z-axis component in the body frame, we can easily coincide the COM with the centroid of the SP.

Fig. 4 shows simulation result done by V-REP simulator [19] when the robot was commanded to coincide its COM (black circles) with the centroid (orange circles) of the SP (red lines) formed by three support feet (red circles) while the other leg is about to swing. Due to the foot slippage during the simulation, the COM get close to the centroid but not coincide exactly as shown in Fig. 4-(a). However, this closeness is still enough to maintain a static stability when the robot is commanded to walk or use its leg as a manipulator.

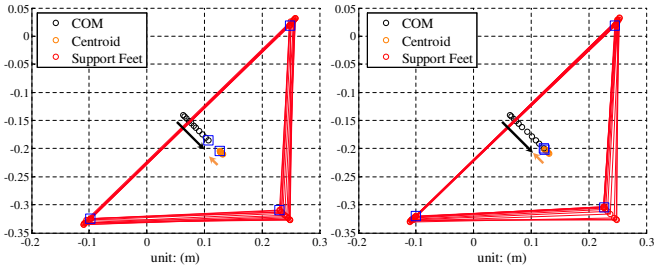


Fig. 4: The robot was commanded to coincide its COM to the centroid formed by the SP in simulation. The final positions of COM, centroid, and supporting feet are marked with the blue squares. (a): by letting $\boldsymbol{P}_{S,d} = [0, 0, z_S]^T$, and (b): by letting $\boldsymbol{P}_{S,d} = [\delta x, \delta y, z_S]^T$.

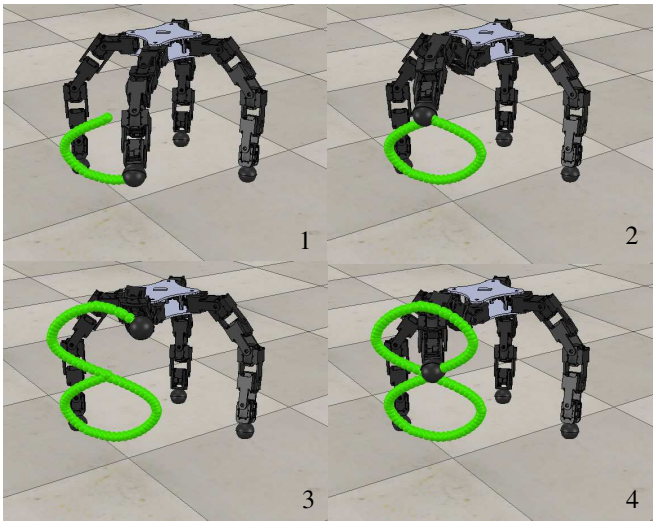
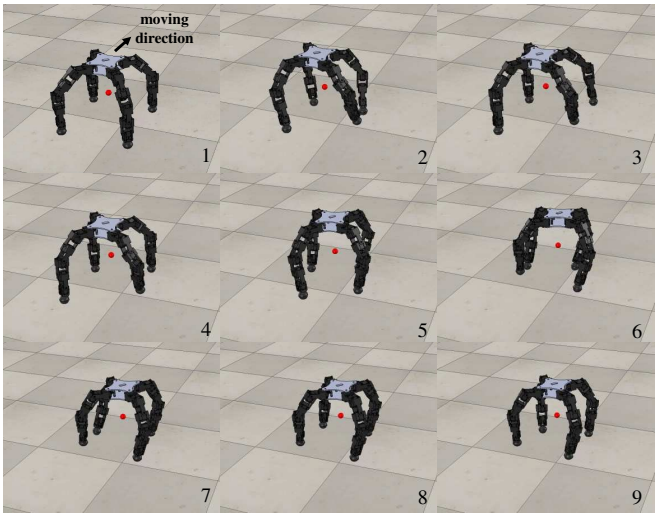
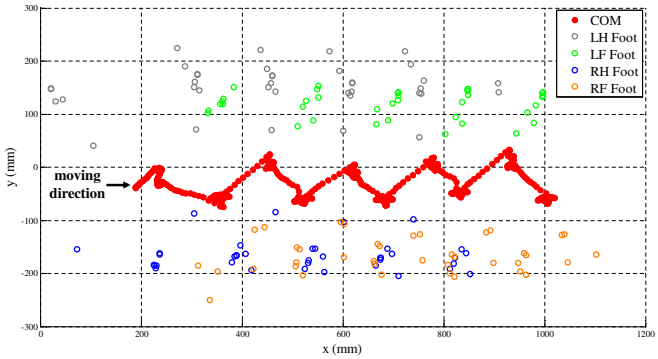


Fig. 5: The robot was commanded to draw a figure eight by using one of its legs as a manipulator in V-REP simulation.



(a) Snapshots of the robot during the simulation. Location of the COM was expressed concurrently with the red sphere. (1): initial posture (2-3): RH leg swing, (4-5): RF leg swing, (6-7): LH leg swing, and (8-9): LF leg swing.



(b) Locations of the robot's COM and the feet when it walked five steps.

Fig. 6: V-REP simulation results when the robot was commanded to walk on the plain terrain.

If we want to strictly coincide the COM with the centroid, this can be accomplished by introducing small offsets δx and δy to the $\boldsymbol{P}_{S,d} = [0, 0, z_S]^T$. Then, by letting desired $\boldsymbol{P}_S$ as $\boldsymbol{P}_{S,d} = [\delta x, \delta y, z_S]^T$ the COM can be coincided exactly with the centroid as shown in Fig. 4-(b).

IV. Simulation Results

To verify the functionality of the proposed robot and the algorithms, V-REP simulator was used in this work. First, we simulated the robot to draw a figure eight as shown in Fig. 5 by considering a situation that the robot uses its leg as a manipulator. Before raising the leg, we commanded the robot to move its COM inside the support polygon formed by the remaining three feet as in Fig. 4. After that the robot started to draw a figure eight in statically stable posture using IJP algorithm, and the root-mean-square error was only 1 mm.

According to [20] incorporating a sideway-swaying (or sway) motion by deliberately translating a robot's body in lateral direction can increase the stability while it walks. Inspired by this idea we also incorporated a sway motion to the proposed robot when simulating a walking. As shown in Fig. 6a, we let the robot to firstly swing the right-hind (RH) leg in take2-3, right-forward (RF) leg in take4-5, left-hind (LH) leg in take6-7, and left-forward (LF) leg in take8-9 sequentially within a single step. At the same time, every time the robot tried to swing its leg, the robot coincided its COM to the centroid formed by the three supporting feet using (4), and this induced the translation of the body in lateral direction diagonally.

Locations of the COM and the four feet viewed from the top are depicted in Fig. 6b when the robot walked 5 consecutive steps in x-axis during the simulation. The COM (the red filled circle) was moved in diagonal direction repeatedly, and always resided in the support polygon. As a result, the robot was able to achieve a statically stable gait.

Also, the robot could cross over an 15 cm height of obstacle in the simulation as shown in Fig. 7. The robot located its COM close to the centroid of the support polygon before swing the leg over the obstacle. From the take1 to take3, the robot swung the right-forward, left-forward, and right-hind leg successively over the obstacle. After that, the robot rotated its body in take4, and swung the left-hind leg over during the take5 to take6. Lastly, the case when the

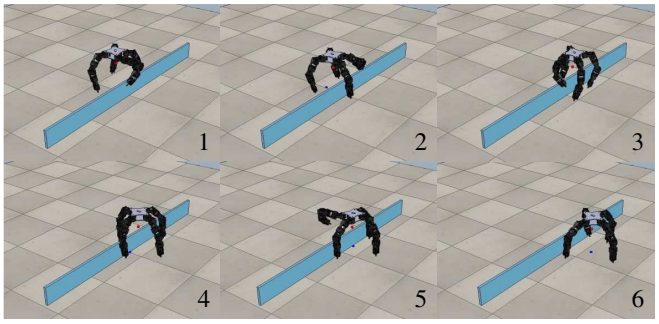


Fig. 7: The robot could overcome an 15 cm height of obstacle in the simulation. (1): RF swing phase, (2): LF swing phase, (3): RH swing phase, and (4-6): LH swing phase.

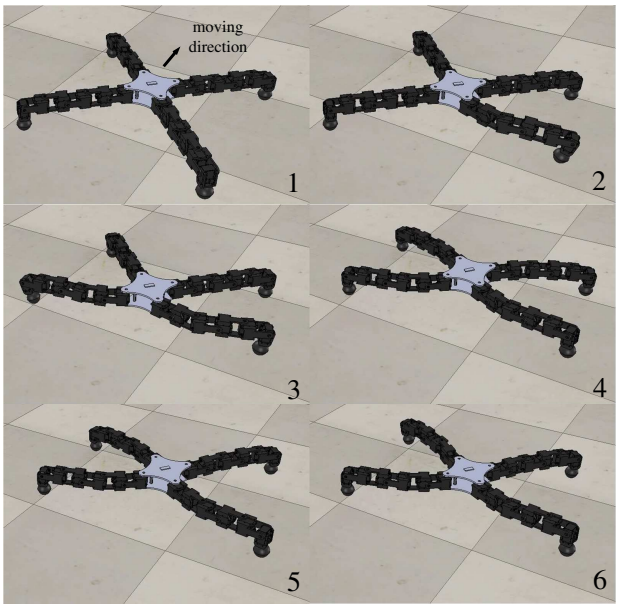


Fig. 8: The robot tried to walk by maintaining its body height as low as possible. 1) initial posture, 2) swing the RH, 3) swing the LH, 4) shift the body forward, 5) swing the RF, and 6) swing the LF.

robot is passing through a flat gap was also considered in the simulation. To make the height of the body as low as possible, the robot stretched out all of its legs except for the last 6^{th} joint on each leg as shown in take1 in Fig. 8. After that the robot swung the right-hind leg, and left-hind leg respectively from the take2 to take3 using IJP algorithm. In take4, the robot shifted its body forward by using (4) in slightly different manner. Previously, when considering P_{Cent} in (3), only a centroid formed by three supporting legs were considered. However, all the four legs were considered to calculate the position of centroid when using (3) only for this case to make a smooth forward translation of the body. In the last phase, the right-forward leg, and the left-forward leg swung forward in the take5 and take6 respectively.

V. Experiment Results

In this section, the robot in Fig. 1 was tested to verify the feasibility of the conducted simulations in Sec.IV. Throughout the whole experiments, the robot was connected with an external power source and a computer. The computer pre-calculated all the desired joint positions and transmitted to the robot through a serial communication. Then, the robot's on-board controller stored the given desired joint positions, and actuated the motors (ROBOTIS DYNAMIXEL, RX-24F [21]). First, the robot was commanded to draw a figure eight as in Fig. 5 by giving the desired joint positions to the proposed robot. The robot located its COM to the centroid of the support polygon, and started actuating its leg as shown in Fig. 9. The robot could draw a figure eight less than four seconds, and this test showed that the robot can do a simple task by using its leg as a manipulator.

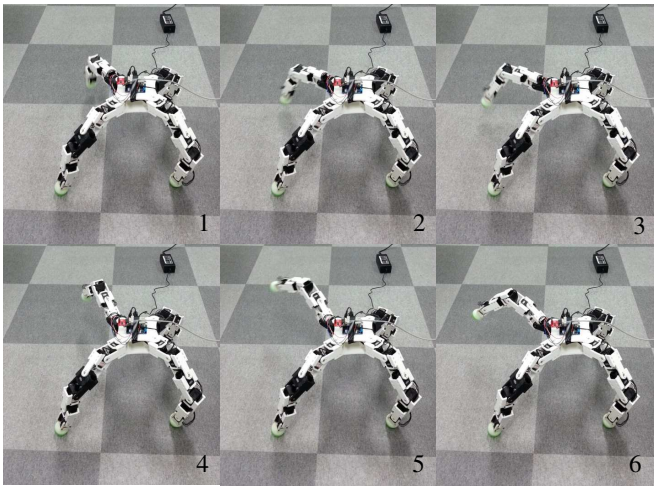


Fig. 9: The robot drew a figure eight as in Fig. 5.

The proposed robot's walking capability was also verified in the experiment as shown in Fig. 10. The robot shifted the COM inside the support polygon formed by the three supporting feet, and swung the right-hind leg from the take1 to take2. Similarly, the robot shifted the COM and swung the right-forward, left-hind, and left-forward leg during the take3-4, 5-6, and 7-8 respectively. The robot could traverse 12 cm forward in a single cycle of walking, and achieve a statically stable gait using (1), and (4).

Lastly, the robot was tested to walk by maintaining its body height as low as possible by stretching out all its legs in Fig. 11. However, the motors' torque were not stronger enough to support the weight of the robot than we expected, and the body almost touched the ground during the experiment. Furthermore, some motors tended to overheat quickly due to the externally applied moment from the ground contact points. Hence, we temporary added a 3 cm height of supporter at the bottom of the robot's body to prevent the motors from over-heating. In Fig. 11, the robot swung the right-hind, left-hind leg, and shifted 8 cm forward during the take1 to take4. After that, the robot swung the two forward legs in the take5 and take6 respectively.

VI. Conclusion and Future Work

In this paper, a redundant quadruped robot that has 24 degrees-of-freedom was proposed for various types of lo-

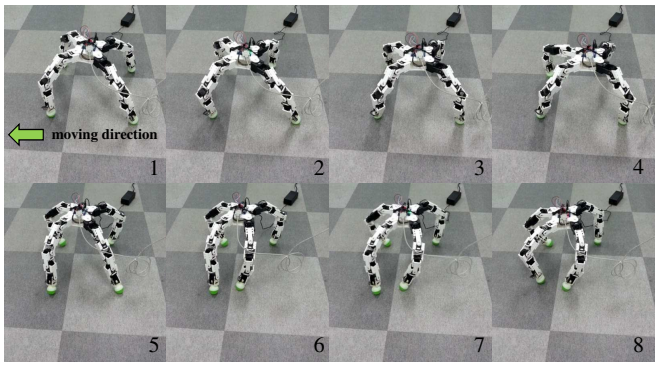


Fig. 10: The robot took a step forward as in Fig. 6a. (1-2): RH swing, (3-4): RF swing, (5-6): LH swing, and (7-8): LF swing.

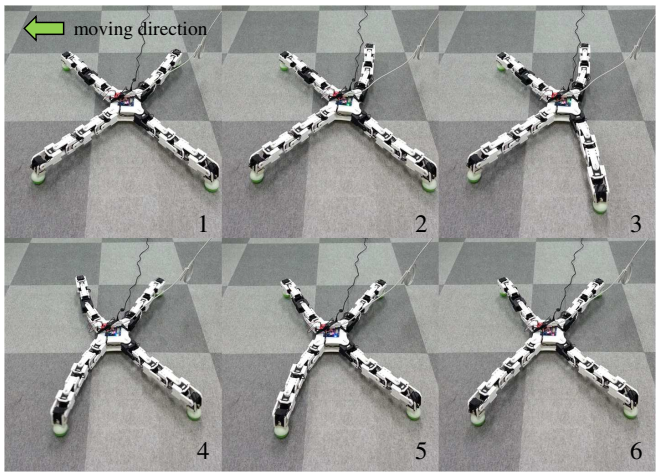


Fig. 11: By maintaining the lowest body height, the robot could walk as in Fig. 8. 1) initial posture, 2) RH swing, 3) LH swing, 4) forward shift, 5) RF swing, and 6) LF swing.

comotion and manipulation. Inverse kinematics of each leg, which has 6 joints, was handled by the improved Jacobian pseudoinverse (IJP) algorithm, which can generate an accurate trajectory even when the initial posture of the leg is close to a singularity. This IJP algorithm was mostly used to generate a desired trajectory of the leg during the walking or using its leg as a manipulator.

In addition, to maintain a statically stable posture when the robot walks or crosses over an obstacle, a combined Jacobian of a center of mass, and a centroid of a support polygon was proposed. This Jacobian helped the robot to walk by diagonally swaying its center of mass, and prevented the robot from falling down while it used its leg as a manipulator or crossed over an obstacle.

All these scenarios were tested in the V-REP simulator and the experiments in open-loop manner. But, the case when the robot crosses over an obstacle was hard to test in real environment without aids of sensor feedback. Hence, we will add some sensors (e.g., gyro sensor, force sensor) in the robot to boost the performance in more complex environment. Additionally, the computational speed of the controller board will be increased for fast response time, and the usage of more powerful motors will be considered as a future works.

REFERENCES

[1] S. Chiaverini, G. Oriolo, and I. D. Walker, *Kinematically Redundant Manipulators. Springer Handbook of Robotics (ed. Bruno Siciliano and Oussama Khatib)*. Springer, 2008.

[2] T. F. Chan and R. V. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *IEEE Transactions on Robotics and Automation*, vol. 11, pp. 286–292, 1995.

[3] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE Journal of Robotics and Automation*, vol. 4, pp. 403–410, 1988.

[4] B. Siciliano and S. Jean-Jacques E, "A general framework for managing multiple tasks in highly redundant robotic system,," in *International Conference on Advanced Robotics*, 1991, pp. 1211–1216.

[5] A. Colome and C. Torras, "Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements," *IEEE/ASME Transactions on Mechatronics*, vol. 20, pp. 944–955, 2015.

[6] A. A. Goldenberg, B. Benhabib, and R. G. Fenton, "A complete generalized solution to the inverse kinematics of robots," *IEEE Journal of Robotics and Automation*, vol. RA-1, pp. 14–20, 1985.

[7] D. Omrcen, L. Zlajpah, and B. Nemec, "Compensation of velocity and/or acceleration joint saturation applied to redundant manipulator," *Robotics and Autonomous Systems*, vol. 55, pp. 337–344, 2007.

[8] F. Flacco, A. D. Luca, and O. Khatib, "Motion control of redundant robots under joint constraints: Saturation in the null space,," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 285–292.

[9] A. Shkolnik and R. Tedrake, "Inverse kinematics for a point-foot quadruped robot with dynamic redundancy resolution,," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 4331–4336.

[10] D. Pongas, M. Mistry, and S. Schaal, "A robust quadruped walking gait for traversing rough terrain,," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1474–1479.

[11] K. Byl, M. Byl, and B. Satzinger, "Algorithmic optimization of inverse kinematics tables for high degree-of-freedom limbs,," in *Proceedings of the ASME Dynamic Systems and Control Conference*, 2014, pp. 1–10.

[12] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Transactions on Man-Machine Systems*, vol. MMS-10, pp. 47–53, 1969.

[13] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modeling, Planning and Control*. Springer, 2009.

[14] A. S. Deo and I. D. Walker, "Minimum effort inverse kinematics for redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, pp. 767–775, 1997.

[15] A. Deo and I. Walker, "Robot subtask performance with singularity robustness using optimal damped least-squares,," in *IEEE International Conference on Robotics and Automation*, 1992, pp. 434–441.

[16] J. R. Rebula, P. D. Neuhaus, B. V. Bonnlander, M. J. Johnson, and J. E. Pratt, "A controller for the littledog quadruped walking on rough terrain,," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1467–1473.

[17] M. Vukobratovic and B. Borovac, "Zero-moment point: Thirty five years of its life," *Internatioanl Journal of Humanoid Robotics*, vol. 1, pp. 157–173, 2004.

[18] K. Byl and M. Byl, "Design of fast walking with one- versus two-at-a-time swing leg motions for robosimian,," in *IEEE International Conference on Technologies for Practical Robot Applications*, 2015, pp. 1–7.

[19] Coppelia Robotics. (2016) V-rep. [Online]. Available: http://www.coppeliarobotics.com/

[20] K. Yoneda and S. Hirose, "Dynamic and static fusion gait of a quadruped walking vehicle on a winding path,," in *IEEE International Conference on Robotics and Automation*, 1992, pp. 143–148.

[21] ROBOTIS. (2016) Dynamicxel. [Online]. Available: http://www.robotis.com/