Received 6 November 2023, accepted 9 November 2023, date of publication 16 November 2023, date of current version 27 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3333388

RESEARCH ARTICLE

Neural Network-Based Joint Velocity Estimation Method for Improving Robot Control Performance

DONGWHAN KIM[1,2], (Graduate Student Member, IEEE), SOONWOOK HWANG[3], MYOTAEG LIM[2], YONGHWAN OH[1], AND YISOO LEE[1], (Member, IEEE)
[1]Korea Institute of Science and Technology (KIST), Seoul 02792, South Korea
[2]School of Electrical Engineering, Korea University, Seoul 02841, South Korea
[3]Environment and Safety Research Center, Samsung Electronics, Hwaseong 18448, South Korea

Corresponding author: Yisoo Lee (yisoo.lee@kist.re.kr)

This work was supported by the Korea Institute of Science and Technology Institutional Programs under Grant 2E32293 and Grant 2E32304.

ABSTRACT Joint velocity estimation is one of the essential properties that implement for accurate robot motion control. Although conventional approaches such as numerical differentiation of position measurements and model-based observers exhibit feasible performance for velocity estimation, instability can be occurred because of phase lag or model inaccuracy. This study proposes a model-free approach that can estimate the velocity with less phase lag by batch training of a neural network with pre-collected encoder measurements. By learning a weighted moving average, the proposed method successfully estimates the velocity with less latency imposed by the noise attenuation compared to the conventional methods. Practical experiments with two robot platforms with high degrees of freedom are conducted to validate the effectiveness of the proposed method.

INDEX TERMS Robotics, robot control, neural network, machine learning, state estimation.

I. INTRODUCTION

In recent years, machine learning has made remarkable advances and is being applied to various fields of engineering. One of the most prominent areas of research is robotics, which is making significant progress [1], [2], [3], [4], [5], [6]. However, machine learning is conservatively applied in robot control applications since stability, one of the most important characteristics of the controller, is hard to guarantee. Accordingly, recent studies utilize machine learning to partially replace a part of the conventional control process to improve performance without adversely affecting stability [7], [8], [9], [10], [11], [12], [13]. In a robot controller, joint velocity estimation results can cause instability of the feedback controller such as proportional-integral-derivative (PID) and proportional-derivative (PD) controller. The instability of a single actuator can adversely affect the whole system since most robots have high degrees of freedom (DoFs) where the joints are dynamically coupled. Moreover, many modern robots (e.g., co-bots, humanoid robots, quadruped robots, etc.) utilize PD-based control schemes which heavily rely on joint velocity estimation accuracy [14], [15], [16]. In addition, estimating the velocity in slow motion is more challenging [17] while the task space motion of the articulated robot has a high probability that slow motion occurs in several joints. However, velocity estimation is still considered a challenging task although there have been related studies for decades [16], [18], [19], [20], [21], [22]. According to the significance, there are still demands for a reliable robot joint velocity estimator. One of the most well-known approaches to estimating joint velocity is based on the numerical differentiation of position measurements with the encoder attached to each actuator of robot joints. It can be extrapolated by dividing the difference between the last two sampled positions by the sampling period [23], [24], [25], [26], [27], [28]. This approach mostly requires

The associate editor coordinating the review of this manuscript and approving it for publication was Thomas Canhao Xu.

filters such as a low-pass filter [29], [30], [31] or moving average filter [32], [33] to attenuate the noise amplified by the differentiation of the encoder measurements. As a price for the noise attenuation, phase lag is imposed by the filtering process, which may cause control instability when the feedback gain increases to improve the accuracy of the controller. A recent study has proposed a high-pass rate limiter to implement a lag-free filtering process [16]. However, the effectiveness of this approach has been limited by the assumption that each joint of the robot precisely tracks the planned trajectory.

Another approach for velocity estimation is model-based methods. Velocity observer [34], [35], [36], [37], [38] is one of the most popular model-based approaches. It computes the expected velocity response based on the encoder measurement, robot model, and control input. When the model is accurately designed, it can estimate the velocity with a small phase lag. In addition, it can reflect the effect of external disturbances. The Kalman filter also similarly operates as a velocity observer [20], [39], [40]. However, these methods are sensitive to the accuracy of the model.

There are several interesting attempts to estimate the velocity through machine learning and neural network. Previously developed methods [19], [41], [42], [43], [44] estimate velocity and acceleration with a neural network that is responsible for differentiation based on the encoder signal. However, these methods are for imitating conventional velocity estimators with a reduced computational burden [19], [42], [43], [44] or for imitating in the sensorless system [45], or they can only be applied for limited motion that they are learned by the data set with specific trajectories [41]. To the best of our knowledge, however, a general approach with a learning-based method that can provide better velocity estimation than conventional non-learning-based methods cannot be found.

In this study, we developed a model-free joint velocity estimator based on the neural network noting the potential of the learning-based method. Although only the present and past velocities are applied as inputs for a neural network, noise filtering results of a small phase lag are achieved thanks to offline learning of the weighted moving average that includes a data set including future information. To mitigate overfitting and overlearning, a combined approach involving cross-validation and early stopping techniques is implemented. Additionally, if the dataset proves insufficient for comprehensive validation, this limitation can be readily addressed. The dataset solely comprises encoder joint velocity data, facilitating expedient and straightforward acquisition. As a result, the proposed method has the following advantages:

- it can improve the control performance of the feedback controller by alleviating the limitations of conventional model-free methods, i.e., phase lag and amplitude loss,
- there is no burden on the modeling since it requires only encoder signal measurements, and
- it is easy to collect data sets and train the neural network.

Due to these advantages, our method makes a crucial contribution in the practical aspect while it successfully enhances the control performance of multi-DoFs robots. To the best of our knowledge, no machine learning-based velocity estimator has been proposed that can improve the control performance of multi-DoF robots better than the proposed method. In this light, the proposed method represents a novel approach to learning-based joint velocity estimation.

The performance and effectiveness of the proposed method are verified through various real-robot control experiments with comparisons. The proposed method is simple and straightforward yet it shows noticeable improvements in the robot feedback control performance. A detailed description of the proposed method is described in Section II, and analyses of the experimental results are shown in Section III. Finally, the conclusions and future work are summarized in Section IV.

II. PROPOSED METHOD

A. LEARNING & NEURAL NETWORK

A reflection of expected future joint behavior can be an effective approach to overcome the model-free approach limitations, i.e., phase lag and amplitude loss. From this point of view, we developed a neural network that can compute the velocity including the influence from the expected future velocity in addition to the past and present information. Although not precise, the tendency of future joint behavior can be predicted by observing the pattern of encoder measurements for a short time horizon from the past to the present. However, since it is difficult to estimate future behavior by analyzing noisy encoder measurements, a learning method is utilized in this study. A neural network is constructed and it is trained to compute the weighted moving average (WMA). The WMA is selected to satisfy the aforementioned considerations since WMA-filtered velocity can include all the past, present, and future information.

For the given K-th sampling data, the WMA-filtered actuator velocity $\dot{\theta}_K^{wma}$ is expressed as follows:

$$\dot{\theta}_K^{wma} = \frac{\sum_{l=-N}^{N} w_{K+l}\dot{\theta}_{K+l}}{\sum_{l=-N}^{N} w_{K+l}}, \tag{1}$$

where N is the number of data points in the past and future observed based on the current sampling time, $w_\bullet$ is the non-negative weighting coefficient for $\bullet$-th sampling data, and $\dot{\theta}_\bullet$ is the input velocity at $\bullet$-th sampling data. In this study, $\dot{\theta}_\bullet$ is computed by numerical differentiation of the encoder measurements. As a result, the above equation describes $2N+1$ data points. In order to increase the influence of the K-th sampling velocity, which is the measured velocity at the reference time, the weighting w_{K+l} is set to be larger when l is closer to zero.

A neural network is then trained to learn the WMA offline. The weights and biases for the neural network are determined through learning with a linear regression approach. The data set is obtained by random joint motion while the labels are calculated by (1). For the learning of a fully connected neural network, Keras Tensorflow is utilized with mean square error loss function and the Adam for the optimizer. ReLU (Rected Linear Unit) is adopted as an activation function while dense is used in the process of deriving output.

A neural network is designed to have $N + 1$ input nodes, two hidden layers with 100 hidden nodes each, and a single output. The number of hidden layers and nodes is empirically determined as the minimum number that can provide sufficient performance. The number of input nodes is determined to have the same number as the past and present data for the learned WMA. The number N is determined as

$$N = \frac{t_w}{\Delta T} - 1, \tag{2}$$

where t_w is time horizon length, and ΔT is sampling time period of real-time system. In this study, t_w is empirically determined as 0.02 s since it showed the best performance. For example, when the frequency of the real-time system is 1 kHz, $N = 19$, and the number of input nodes is 20 accordingly. Note that the number N in WMA equation (1) for labeling has to be the same as N of the neural network. So, 39 data points are used to calculate (1) when $t_w = 0.02$. The output node then generates the velocity result.

The inputs of the proposed method consist of numerically calculated velocities instead of position measurements from the encoder. This choice simplifies the constructed neural network and facilitates easier training. The additional preprocessing step for numerical differentiation incurs a small cost compared to the benefits it offers. Moreover, the preprocessing only requires computation for numerical differentiation, which is relatively straightforward. It is worth noting that employing a neural network with measured position inputs to achieve similar results as the proposed method may necessitate more extensive training data sets, a more complex neural network structure, and a larger number of hidden layers and nodes, thereby compromising practicality. The trained neural network introduced in the previous subsection additionally requires preprocessing and post-processing. The overview of the entire process is shown in Fig. 1.

B. ESTIMATION PROCESS

1) PREPROCESS

The velocity of the actuator is numerically computed as the developed neural network requires velocity for input nodes. The velocity can be simply calculated by 1st-order approximation with the encoder measurement as follows:

$$\dot{\theta}_K = \frac{\theta_K - \theta_{K-1}}{\Delta T}, \tag{3}$$

where $\theta_\bullet$ is the position of the actuator measured by encoder at $\bullet$-th sampling data. The above equation can be replaced with more advanced numerical differentiation methods such as the method in [23]. Here, the input velocity equation for training and for the neural network should be the same. To stack the present and past N data for neural network computation, the calculated velocity is then recorded in the buffer memory.

2) NEURAL NETWORK COMPUTATION

For neural network computation, the velocities in the buffer are inserted into the input nodes. Then, the velocity is calculated by computing the neural network introduced in the previous subsection with nodes having weights and bias parameters determined by offline learning.

3) POST PROCESS

The output of the neural network $\dot{\theta}_K^{nn}$ is the estimated actuator velocity. However, $\dot{\theta}_K^{nn}$ might not be able to describe the rapid velocity change since the time interval t_w of WMA learned by the neural network is relatively long. Therefore, we modify $\dot{\theta}_K^{nn}$ by combining it with the velocity $\dot{\theta}_K$ calculated by (3) at the same sampling time to regenerate high-frequency response data.

$$\dot{\theta}_K^{est} = (1-\alpha)\dot{\theta}_K^{nn} + \alpha\dot{\theta}_K, \quad (0 < \alpha < 1) \tag{4}$$

where α is a coefficient. When α becomes small, the velocity can be smoothened while it cannot properly describe rapid motion. However, it could be operated more effectively in slow motion. In this study, α is set to 0.2. For the joint with gear, estimated joint velocity $\dot{q}_K^{est}$ can be calculated as follows:

$$\dot{q}_K^{est} = \frac{1}{N_{gear}}\dot{\theta}_K^{est}, \tag{5}$$

where N_{gear} is the gear ratio.

III. EXPERIMENTAL VERIFICATION

A. ANALYSIS OF PROPOSED METHOD

In this subsection, the results of the proposed method are analyzed to observe the characteristics of the proposed method. First, the proposed velocity estimator is applied to a known sine wave signal with Gaussian noise to clearly compare with the ground truth data and the estimation result. Data sets with a constant frequency and magnitude similar to a sine wave are not used for learning to prevent over-fitting. As can be seen in Fig. 2, high-frequency noise is drastically reduced with a small delay compared to the reference data that describes ground truth.

In addition, comparative experiments are conducted to observe the effect and performance of the proposed method in an actual actuator. The actuator is composed of a motor, 100:1 harmonic gear, and 2500 ppr optical encoder. In addition to the proposed method, the velocity calculated by the numerical differentiation of the measured encoder signal, the result of applying a first-order low-pass filter with 250 Hz cutoff frequency to it, and the velocity observer [34] are also implemented for the comparison. The cutoff frequency of the low-pass filter is determined to generate a similar magnitude to that of the proposed method. As can be seen in Fig. 3,

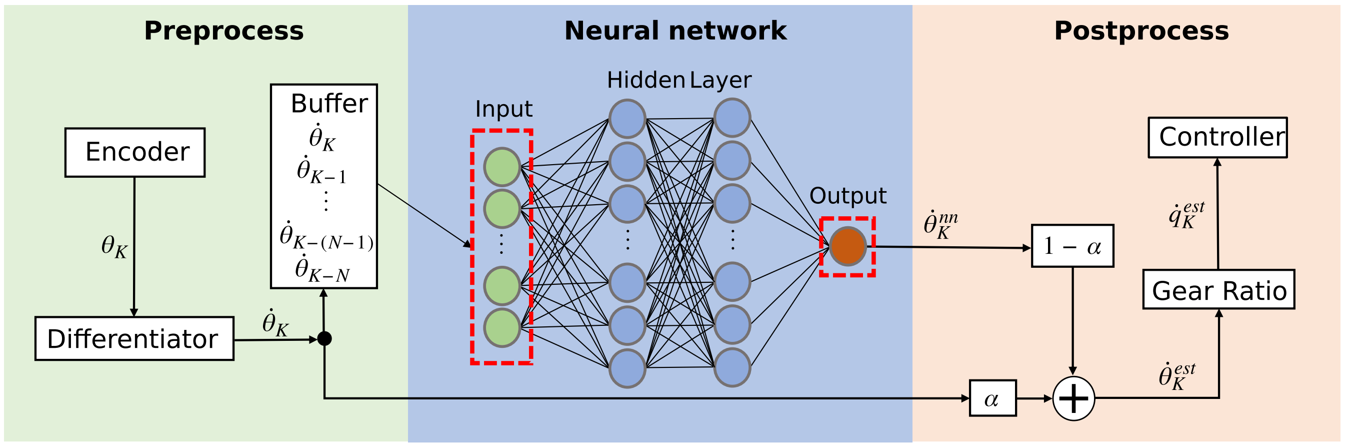


FIGURE 1. Structure of the proposed velocity estimation method.

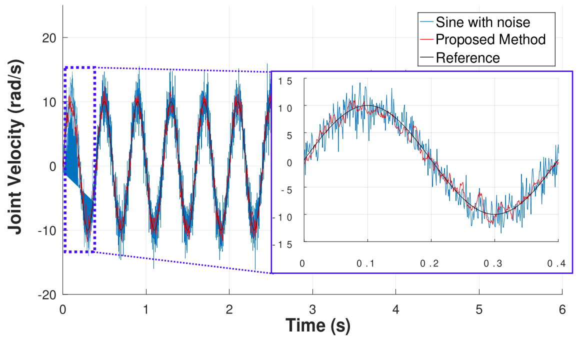


FIGURE 2. Velocity estimation result of the proposed method for a sine wave with Gaussian noise. A magnified view in the plot shows details for a wavelength.

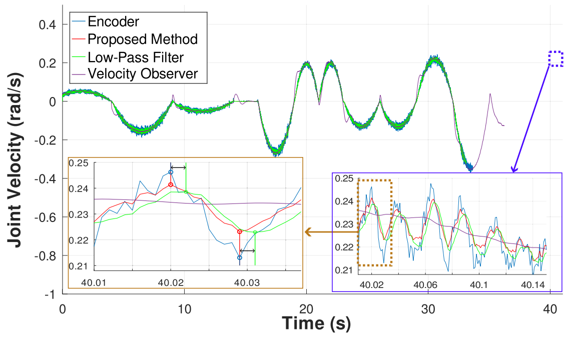


FIGURE 3. Comparisons of velocity estimation results of real robot actuator.

compared to the encoder measurements, the proposed method presented in the red-colored line exhibits minimal delay and effectively diminishes vibrations resulting from noise. In contrast, the green-colored line representing the low-pass filter demonstrates a reduction in magnitude compared to the encoder measurements and introduces a notable delay. The purple-colored line, representing the velocity observer result, is a model-based method. It does not capture the motor's actual driving characteristics and incorporates a robust smoothing process, which results in a distinctive trend.

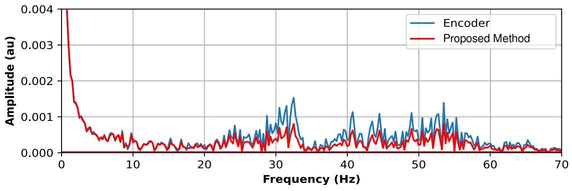


FIGURE 4. Fast Fourier transform results of the velocity numerically extrapolated.

The phase lag can be approximately known by comparing the peak point of each phase. The proposed method has a phase lag of less than 0.001 s while the low-pass filter has approximately 0.02-0.03 s phase lag. The velocity observer which is a model-based approach has a large difference from the other results. From the result, one can notice that the proposed method generates the intended effect. It can estimate the noise-attenuated velocity with less phase lag and less amplitude loss compared to the filtered result.

In order to analyze the filtering effect that the proposed method generates, a fast Fourier transform (FFT) is performed on the actual actuator experimental result. As shown in Fig. 4, the blue-colored line representing the encoder measurements exhibits substantial high-frequency noise above 20 Hz. In contrast, the red-colored line, which represents the proposed method, clearly demonstrates a significant reduction in noise beyond 20 Hz.

B. EXPERIMENTS

1) ROBOT SETUP

Real robot experiments are conducted to evaluate how much the proposed velocity estimator can contribute to improving the performance of various types of robot controllers.

Two robotic platforms, a 14-DoFs dual-arm manipulator shown in Fig. 5 and a 33-DoFs humanoid robot TOCABI [46] shown in Fig. 6 are utilized for the experiments.

The feedback control tasks are implemented by varying the joint velocity estimation methods for comparison. For this

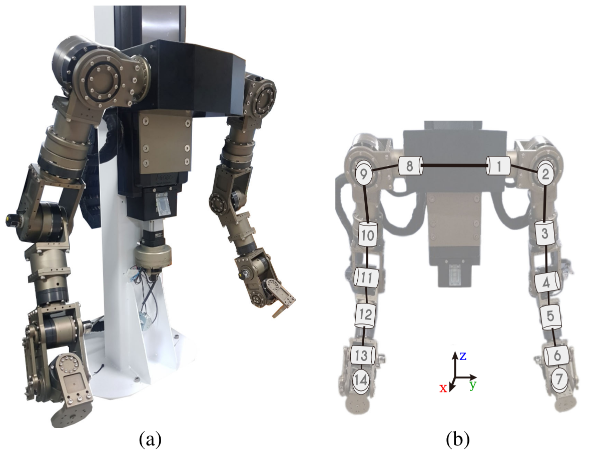


FIGURE 5. (a) Shape of a dual-arm manipulator utilized for experiments, and (b) schematic kinematic structure and joint number of the robot.

FIGURE 6. Shape of a humanoid robot TOCABI utilized for experiment.

purpose, robot dynamics-based joint space and task space controllers [47] are performed in the dual-arm manipulator, and the whole-body controller [48] is performed in the humanoid robot. Each joint of both robots is composed of a brushless DC electric motor (BLDC motor), 100:1 harmonic gear, and encoder. As for the motors of both robots, several different motors from the same company (Kollmorgen) are used. Motor current control-based joint torque control is performed by the motor controller under the assumption that input current to the motor is linearly proportional to output torque. The desired torque is computed in the computer as the high-level controller while communication is conducted with EtherCAT. The encoders of the dual-arm manipulator are 2500 ppr (pulse per revolution) optical encoder, and of the humanoid are 13-bit magnetic incremental encoder. The control frequency of the dual-arm and humanoid are 1 kHz and 2 kHz, respectively. For the velocity estimation, a single neural network is trained for each robot, and each joint velocity is computed through the neural network in real-time. For fast computation, the neural network is programmed in the computer with C++.

2) DATA COLLECTION AND TRAINING

To train the developed neural network, random motions of all the joints of a robot are generated by controlling the joint positions. The goal position and time for each motion are randomly selected to obtain various velocity profiles. In order to avoid joint limits or self-collision during the random motion, the boundaries of goal position and time are limited to a conservatively determined narrow range. As learning is only affected by joint velocity, the learning can be performed well even with narrow-range position constraints. During the motion, velocities computed by (3) of all joints are recorded. Then, WMA expressed in (1) is computed and applied for offline training.

In order to collect data in a time-efficient manner, we generated motions of all the joints together and trained a single neural network with the result data. As a result, learning a neural network for each robot is completed with the data collected from the random motion within 10 min. In this study, the neural network of the dual-arm manipulator and humanoid are trained separately with the data obtained from the joints of each robot. For the learning, a batch size of 2048, the number of epochs 4000, Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a learning rate 0.001 are used for optimization. To avoid over-fitting, 80% of the random motion data is used for training and 20% for the testing set.

3) COMPUTATION TIME

For a real-time control system, computation time to estimate the velocity of all the joints should take less than the sampling time period ΔT. Since several calculations for control are required during ΔT, it is advantageous to shorten the calculation time required for the velocity estimation. The computation time is measured in the computer with Intel Core i7-12700F CPU 4.9 GHz boost clocks with 32 GB memory. The measured computation time includes the whole process described in Fig. 1 for 14 joints. The average computation time of 1 million samples of the proposed method is 0.063 ms. One can notice that the joint velocity estimation does not burden the real-time calculation for robots in this section since ΔT of robots are 1 ms and 0.5 ms. Therefore, the proposed method has a very cheap computational cost that can be applied in most robotic systems.

In the real-time control system, a computation time constraint arises from the number of joints in the robot system. As the proposed neural network has to be computed as many times as the total number of joints, the computational time burden increases with the number of joints, potentially posing challenges for real-time control. However, multithreading-based parallel computing can be utilized as each estimate of joint velocity is independent of the other joints, so all joint velocity estimations are concurrently processed with a single computational cost (0.063 ms on average) in the experimented robots. It is worth noting that the entire operation is completed within 0.1 ms to successfully ensure 1 kHz to 2 kHz real-time. Therefore, the experiments conducted do not result in lag or delay caused by joint velocity operations.

4) JOINT SPACE CONTROL

The joint space control task is performed by a dual-arm manipulator. To control the motions of joints, the following dynamics-based joint space PD-control equation is utilized:

$$\mathbf{\Gamma} = \mathbf{A}\{k_p(\mathbf{q}_d - \mathbf{q}) + k_v(\dot{\mathbf{q}}_d - \dot{\mathbf{q}})\} + \mathbf{h}, \tag{6}$$

where $\mathbf{\Gamma}$ is the joint torque vector, $\mathbf{A}$ is the joint space inertia matrix, $\mathbf{q}$ is the joint angle vector, $\mathbf{q}_d$ is the desired joint angle vector, $\mathbf{h}$ is the nonlinear force vector including the Coriolis/centrifugal and gravity force, and k_p, k_v are proportional and derivative gain, respectively.

All the joints are controlled to track the trajectory generated by the cubic spline in the experiment. The joint positions are obtained by the encoder and the joint velocities are estimated by the proposed method. For the comparison, the same task is performed with varying velocity estimators. Numerical differentiation with the 20 Hz cutoff frequency low-pass filter and the velocity observer are applied. The cutoff frequency of the low-pass filter and the maximum control gains are empirically explored and applied to achieve the best control performance. The maximum gains adopted are shown in Table 1.

TABLE 1. Joint space control gain.

	Low-Pass Filter	Velocity Observer	Proposed Method
k_p	300	600	850
k_v	10	60	35

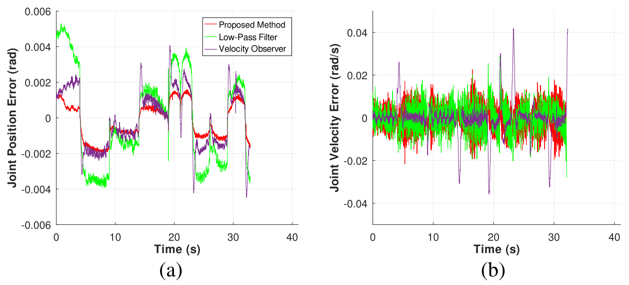


FIGURE 7. Left arm shoulder joint control results: (a) Joint position control errors and (b) joint velocity control errors of a dual-arm manipulator. The red colored line is when the proposed method is applied, the green colored line is when the numerical differentiation with low-pass filter is applied, and the purple line is when the velocity observer is applied.

Comparative experimental results can be seen in Fig. 7 and Table 2. In the figures, the control errors of the shoulder pitch joint of the left arm are represented. Control results of the other joints are not plotted since they have similar tendencies. The best performance is obtained when the proposed method is applied. Both gains are increased and the control errors are reduced proportionally when the proposed method is applied compared to the low-pass filtered result. Although the velocity observer can also apply gains higher than low-pass filtered result, it creates a larger control error than the proposed method. In addition, the velocity from the velocity observer bounces when each motion starts. It is notable that although the D gain k_v of the velocity observer is the largest, the damping effect is smaller than the result of the proposed method. This is presumably because the model for the observer has an error so the difference between the actual and the estimated velocity is large.

5) TASK SPACE CONTROL

The task space control experiment is also performed by the dual-arm manipulator. To control the end-effector motion in the Cartesian space $\mathbf{x}$ of the dual-arm manipulator, the following operational space PD control equation is utilized:

$$\mathbf{\Gamma} = \mathbf{J}^T\mathbf{\Lambda}\{k_p(\mathbf{x}_d - \mathbf{x}) + k_v(\dot{\mathbf{x}}_d - \dot{\mathbf{x}})\} + \mathbf{h}, \tag{7}$$

where $\mathbf{J}$ is the Jacobian matrix which describes the relationship between the task space and the joint space as $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$, $\mathbf{\Lambda} = (\mathbf{J}\mathbf{A}^{-1}\mathbf{J}^T)^{-1}$ is the operational space inertia matrix, and $\mathbf{x}_d$ is the desired position vector at the operational space.

The task space $\mathbf{x}$ is defined to describe 12-DoFs (6-DoFs for each arm) end-effectors Cartesian space position (x, y, z) and orientation (roll, pitch, yaw). The task space velocity $\dot{\mathbf{x}}$ is calculated by the equation $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$ with estimated joint velocities. The trajectory is designed by a cubic spline to move the position and orientation of the end-effector during 5 s. Positions and orientations are commanded to move 0.1 m and 20 deg, respectively for the first 2.5 s; and -0.1 m and -20 deg, respectively for the rest of the time.

For the comparison, the same control task is performed with varying velocity estimators. Numerical differentiation with the low-pass filter, and the velocity observer are applied for velocity estimation. As the maximum control gains change according to the velocity estimator, we empirically tuned gains with the largest number that is stable without divergence of the controller as can be seen in Table 3. As a result, the right arm end-effector control error can be seen in Fig. 8. The left-arm control results are omitted since it has practically the same result. Similar to the joint space control, one can observe that the proposed method can achieve the highest gains for PD control and accordingly, produce the smallest control error. Compared to the low-pass filtered result, the control performance of all the task components tends to increase in proportion as the gains increase. This is presumably because the proposed method has less phase lag and amplitude loss than the low-pass filter. In the case of the velocity observer, the maximum gains are the same as the proposed method and accordingly, the control error magnitude is smaller or similar to the proposed method. However, it is notable that the directions of the velocity observer result in x, y, z, yaw are opposite, which means that they continuously overshoot. This causes the problem of

TABLE 2. Joint space control error.

	Joint	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Joint Position Error (10^{-2}rad, RMS)	Low-Pass Filter	0.27	0.23	0.20	0.28	2.34	3.65	5.96	0.31	0.17	0.27	0.32	3.04	3.38	8.07
	Velocity Observer	0.18	0.13	0.09	0.13	0.98	1.86	2.60	0.20	0.10	0.14	0.15	1.60	1.74	3.90
	Proposed Method	0.11	0.07	0.07	0.09	0.71	1.28	1.90	0.12	0.06	0.09	0.12	1.00	1.22	2.06
Joint Velocity Error (10^{-2}rad/s, RMS)	Low-Pass Filter	0.63	0.57	0.78	0.15	3.20	4.43	9.37	0.71	0.33	0.67	0.21	3.25	3.49	7.03
	Velocity Observer	0.73	0.39	0.45	0.26	1.93	3.03	4.61	0.80	0.29	0.46	0.34	2.34	3.24	4.67
	Proposed Method	0.50	0.27	0.43	0.24	1.94	2.54	5.24	0.67	0.27	0.52	0.31	2.00	2.39	4.15

TABLE 3. Task space control gain.

	Low-Pass Filter	Velocity Observer	Proposed Method
k_p	300	800	800
k_v	10	20	20

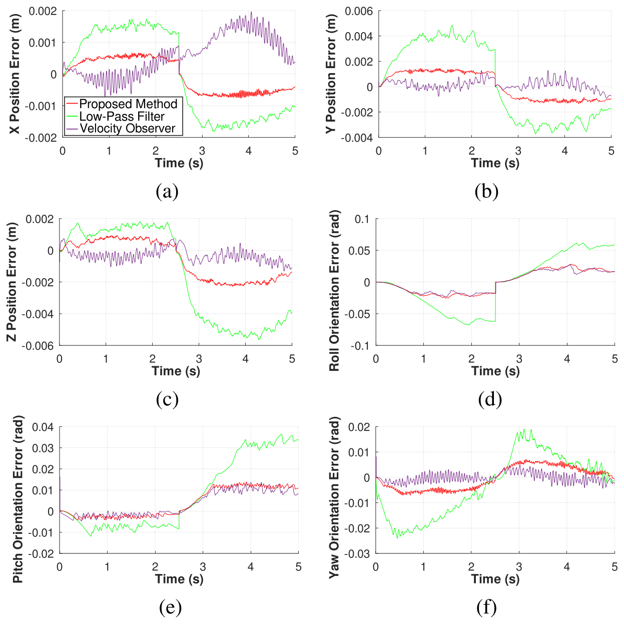


FIGURE 8. Task space control results: control errors of the end-effector position (a) *X*, (b) *Y*, (c) *Z*, and orientation (d) Roll, (e) Pitch, (f) Yaw. The red-colored line is when the proposed method is applied, the green-colored line is when the numerical differentiation with low-pass filter is applied, and the purple line is when the velocity observer is applied.

generating a motion that repeats moving and stopping within a short period of time causing vibration. In addition, the error of the x-axis is larger than the proposed method despite the same gains. This is presumably because the velocity from the velocity observer has a relatively large difference from the actual velocity caused by the modeling error.

6) WHOLE-BODY CONTROL

To verify the proposed method in a more complex system, it is also implemented in a hierarchical quadratic programming-based whole-body controller [48] of a humanoid robot. The pelvis position and orientation are controlled by a PD control scheme as the highest priority task, and the joint angles are controlled to keep posture as the second priority task while the robot dynamics, kinematics, and contact constraints for rigid contact are set as strict constraints.

The trajectory is designed by the quintic spline and the trajectory tracking control is performed during 5 s. During 0-2.5 s, the pelvis is controlled to move 0.12 m along the x-axis, and 0.05 m for the y-axis. During 0.25-5 s, the pelvis is controlled to return to the initial position. The rest of task components are controlled to keep the initial state.

In the humanoid robot controller, joint velocity is estimated by numerical differentiation of the encoder measurement with a 60 Hz cutoff frequency low-pass filter. With the low-pass filtered velocity, the maximum gains k_p and k_v are 80 and 5, respectively; and 200 and 20, respectively with the proposed method. The control results can be seen in Fig. 9. One can observe that the control error is smaller when the proposed method is applied since higher gains can be used. As already mentioned in [16], [49], and [50], due to the influence of large inertia, modeling error, limited control bandwidth, etc., the torque control-based whole-body control has difficulty in increasing the PD control gains, but the proposed method successfully improved. This supports that the estimation of the velocity itself has become more accurate.

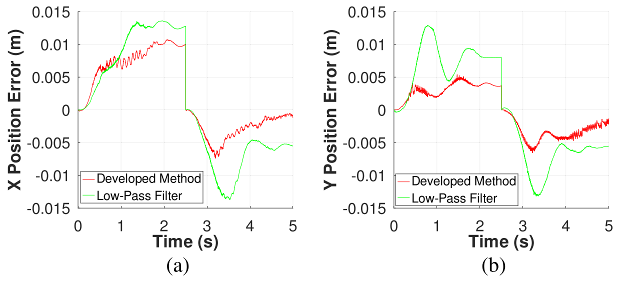


FIGURE 9. Whole-body control results with the humanoid robot TOCABI: control errors of the pelvis position (a) *X* and (b) *Y*. The red-colored line is when the proposed method is applied, and the green-colored line is when the numerical differentiation with a low-pass filter is applied.

C. DISCUSSIONS

The proposed method is a model-free approach that requires encoder measurements as input, and has low dependency on robot systems. The advantage of this approach is verified by showing that velocities of all joints can be estimated by a single trained neural network with noticeable performance improvements, even though the experimental robots have complex dynamics and several different motor products.

TABLE 4. Average control error.

	Joint Space Control (10^{-2} rad)	Task Space Position Control (10^{-3} m)	Task Space Rotation Control (10^{-2} rad)	Pelvis Position Control (10^{-3} m)
Low-Pass Filter	2.035	7.073	6.127	14.590
Proposed Method	0.636	2.745	2.650	8.198

In addition, the proposed method can be utilized in other control schemes such as PID because the developed method does not depend on the controller.

In addition to the performance, we highlight that the proposed method can learn effectively with relatively small data. The consumed time for data collection from the real robot experiment in this study is completed in a short time (less than 10 min). If the motion can be generated effectively to represent various cases rather than random motion, higher performance can be achieved with less data. This is a crucial aspect as many learning-based methods demand substantial effort and trial and error to gather suitable data sets. The challenge is further compounded when real-robot experiment data are necessary, increasing the time and effort required for data collection. This obstacle serves as a barrier to implementing learning-based methods, hindering their reproducibility in other robotic systems. In particular, the proposed method proves highly efficient for robots with multiple joints, as it obviates the need to train separate neural networks for each joint. This advantage streamlines the implementation process and enhances the method's applicability across diverse robotic systems.

Meanwhile, although the number of nodes of the proposed neural network is not large, it has to be carefully implemented if a robot has low computation power and many joints since neural network computation is required for each joint.

IV. CONCLUSION

This study presents a novel model-free learning-based joint velocity estimation method. The effectiveness of the method is analyzed and verified through real-robot experiments. In particular, we showed that it can improve the control performance of various types of robot controllers, i.e., joint controller, task space controller, and whole-body controller. The improvements are clearly shown through comparative experiments with other well-known methods. The proposed method facilitates an enhancement in control gain through precise velocity estimation during PD control, leading to a significant reduction in control error, as evidenced in Table 4. The findings reveal control error reductions of 68.76% in joint space control, 61.19% in task space position control, 56.75% in task space rotation control, and 43.81% in pelvis position control.

In addition, one can notice that machine learning can effectively improve robot control performance when utilized as a state estimator. This approach is differentiated from the conventional neural network-based methods that merely imitate conventional algorithm-based methods. From this point of view, we expect to further improve robot control performance in the future by extending the proposed approach. Accordingly, we will explore a more optimal structure of the neural network that can generate improved performance with less computational time. In addition, a joint acceleration estimation will be developed by advancing the proposed approach. By estimating both velocity and acceleration, the estimation method can be applied to various types of controllers. As a result, the performance of the robot controller can be drastically improved.

ACKNOWLEDGMENT

The authors appreciate Prof. Jaehueng Park, Junewhee Ahn, and Daegyu Lim for conducting humanoid robot experiments.

REFERENCES

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Sci. Robot.*, vol. 4, no. 26, Jan. 2019, Art. no. eaau5872.

[2] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, Sep. 2013.

[3] V. Prabakaran, A. V. Le, P. T. Kyaw, P. Kandasamy, A. Paing, and R. E. Mohan, "STetro-D: A deep learning based autonomous descending-stair cleaning robot," *Eng. Appl. Artif. Intell.*, vol. 120, Apr. 2023, Art. no. 105844.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auto. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[5] M. Maroto-Gómez, S. Marqués-Villaroya, J. C. Castillo, Á. Castro-González, and M. Malfaz, "Active learning based on computer vision and human–robot interaction for the user profiling and behavior personalization of an autonomous social robot," *Eng. Appl. Artif. Intell.*, vol. 117, Jan. 2023, Art. no. 105631.

[6] R. Szczepanski, K. Erwinski, M. Tejer, A. Bereit, and T. Tarczewski, "Optimal scheduling for palletizing task using robotic arm and artificial bee colony algorithm," *Eng. Appl. Artif. Intell.*, vol. 113, Aug. 2022, Art. no. 104976.

[7] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6023–6029.

[8] Y.-C. Chang, N. Roohi, and S. Gao, "Neural Lyapunov control," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[9] W. Zheng, H. Wang, Z. Zhang, and H. Wang, "Adaptive robust finite-time control of mobile robot systems with unmeasurable angular velocity via bioinspired neurodynamics approach," *Eng. Appl. Artif. Intell.*, vol. 82, pp. 330–344, Jun. 2019.

[10] D. Lim, D. Kim, and J. Park, "Momentum observer-based collision detection using LSTM for model uncertainty learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 4516–4522, doi: 10.1109/ICRA48506.2021.9561667.

[11] B. R. O. Floriano, A. N. Vargas, J. Y. Ishihara, and H. C. Ferreira, "Neural-network-based model predictive control for consensus of nonlinear systems," *Eng. Appl. Artif. Intell.*, vol. 116, Nov. 2022, Art. no. 105327.

[12] Z. Xu, S. Li, X. Zhou, S. Zhou, T. Cheng, and Y. Guan, "Dynamic neural networks for motion-force control of redundant manipulators: An optimization perspective," *IEEE Trans. Ind. Electron.*, vol. 68, no. 2, pp. 1525–1536, Feb. 2021.

[13] H. Su, W. Qi, C. Yang, J. Sandoval, G. Ferrigno, and E. D. Momi, "Deep neural network approach in robot tool dynamics identification for bilateral teleoperation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2943–2949, Apr. 2020.

[14] A. Albu-Schäffer, C. Ott, and G. Hirzinger, "A unified passivity-based control framework for position, torque and impedance control of flexible joint robots," *Int. J. Robot. Res.*, vol. 26, no. 1, pp. 23–39, Jan. 2007.

[15] H. Lee and J. Park, "Operational space control under actuator bandwidth limitation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 12847–12852.

[16] J. Englsberger, G. Mesesan, A. Werner, and C. Ott, "Torque-based dynamic walking—A long way from simulation to experiment," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 440–447.

[17] G. Liu, "On velocity estimation using position measurements," in *Proc. Amer. Control Conf.*, Oct. 2002, pp. 1115–1120.

[18] B. Xiao, L. Cao, S. Xu, and L. Liu, "Robust tracking control of robot manipulators with actuator faults and joint velocity measurement uncertainty," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 3, pp. 1354–1365, Jun. 2020.

[19] J. Peng, S. Ding, Z. Yang, and F. Zhang, "Neural network-based hybrid position/force tracking control for robotic systems without velocity measurement," *Neural Process. Lett.*, vol. 51, no. 2, pp. 1125–1144, Apr. 2020.

[20] J. Kirchhoff and O. von Stryk, "Velocity estimation for ultralightweight tendon-driven series elastic robots," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 664–671, Apr. 2018.

[21] S. A. B. Birjandi, J. Kühn, and S. Haddadin, "Joint velocity and acceleration estimation in serial chain rigid body and flexible joint manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 7503–7509.

[22] S. S. Saab and P. Ghanem, "A multivariable stochastic tracking controller for robot manipulators without joint velocities," *IEEE Trans. Autom. Control*, vol. 63, no. 8, pp. 2481–2495, Aug. 2018.

[23] R. H. Brown, S. C. Schneider, and M. G. Mulligan, "Analysis of algorithms for velocity estimation from discrete position versus time data," *IEEE Trans. Ind. Electron.*, vol. 39, no. 1, pp. 11–19, Mar. 1992.

[24] B. Veselic, B. Perunicic-Drazenovic, and Č. Milosavljevic, "Improved discrete-time sliding-mode position control using Euler velocity estimation," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3840–3847, Nov. 2010.

[25] F. Flacco and A. De Luca, "Discrete-time velocity control of redundant robots with acceleration/torque optimization properties," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 5139–5144.

[26] R. H. Jaafar and S. S. Saab, "Approximate differentiator with varying bandwidth for control tracking applications," *IEEE Control Syst. Lett.*, vol. 5, no. 5, pp. 1585–1590, Nov. 2021.

[27] J. Moreno-Valenzuela and L. González-Hernández, "Operational space trajectory tracking control of robot manipulators endowed with a primary controller of synthetic joint velocity," *ISA Trans.*, vol. 50, no. 1, pp. 131–140, Jan. 2011.

[28] R. H. Brown and S. C. Schneider, "Velocity observations from discrete position encoders," *Proc. SPIE*, vol. 858, pp. 1111–1118, Jan. 1987.

[29] R. Volpe and P. Khosla, "A theoretical and experimental investigation of explicit force control strategies for manipulators," *IEEE Trans. Autom. Control*, vol. 38, no. 11, pp. 1634–1650, Nov. 1993.

[30] J. Lee, P. H. Chang, and M. Jin, "Adaptive integral sliding mode control with time-delay estimation for robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 64, no. 8, pp. 6796–6804, Aug. 2017.

[31] W.-H. Zhu and T. Lamarche, "Velocity estimation by using position and acceleration sensors," *IEEE Trans. Ind. Electron.*, vol. 54, no. 5, pp. 2706–2715, Oct. 2007.

[32] J. S. Hunter, "The exponentially weighted moving average," *J. Quality Technol.*, vol. 18, no. 4, pp. 203–210, 1986.

[33] Y. Zhuang, L. Chen, X. S. Wang, and J. Lian, "A weighted moving average-based approach for cleaning sensor data," in *Proc. 27th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2007, p. 38.

[34] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Trans. Robot.*, vol. 33, no. 6, pp. 1292–1312, Dec. 2017.

[35] B. Bona and M. Indri, "Analysis and implementation of observers for robotic manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, Oct. 1998, pp. 3006–3011.

[36] S.-M. Yang and S.-J. Ke, "Performance evaluation of a velocity observer for accurate velocity estimation of servo motor drives," *IEEE Trans. Ind. Appl.*, vol. 36, no. 1, pp. 98–104, Jan. 2000.

[37] W.-H. Chen, D. J. Ballance, P. J. Gawthrop, and J. O'Reilly, "A nonlinear disturbance observer for robotic manipulators," *IEEE Trans. Ind. Electron.*, vol. 47, no. 4, pp. 932–938, Aug. 2000.

[38] F. Bouakrif, D. Boukhetala, and F. Boudjema, "Velocity observer-based iterative learning control for robot manipulators," *Int. J. Syst. Sci.*, vol. 44, no. 2, pp. 214–222, Feb. 2013.

[39] M. Gautier and P. Poignet, "Extended Kalman filtering and weighted least squares dynamic identification of robot," *Control Eng. Pract.*, vol. 9, no. 12, pp. 1361–1372, Dec. 2001.

[40] I.-A. Viorel and H. Hedesiu, "On the induction motors speed estimators robustness against their parameters," in *Proc. IEEE Int. Electr. Mach. Drives Conf.*, Jun. 2001, pp. 931–934.

[41] S. Chan, "Velocity estimation for robot manipulators using neural network," *J. Intell. Robot. Syst.*, vol. 23, no. 2, pp. 147–163, 1998.

[42] A. Goedtel, I. N. Silva, P. J. A. Serni, and C. F. Nascimento, "Neural approach for speed estimation in induction motors," in *Proc. 7th Int. Conf. Intell. Syst. Design Appl. (ISDA)*, Oct. 2007, pp. 561–566.

[43] P. P. Cruz and J. J. R. Rivas, "A small neural network structure application in speed estimation of an induction motor using direct torque control," in *Proc. IEEE 32nd Annu. Power Electron. Spec. Conf.*, Jun. 2001, pp. 823–827.

[44] L. Ben-Brahim, "Motor speed identification via neural networks," *IEEE Ind. Appl. Mag.*, vol. 1, no. 1, pp. 28–32, Oct. 1995.

[45] E. Ilten, H. Calgan, and M. Demirtas, "Design of induction motor speed observer based on long short-term memory," *Neural Comput. Appl.*, vol. 34, no. 21, pp. 18703–18723, Nov. 2022.

[46] M. Schwartz, J. Sim, J. Ahn, S. Hwang, Y. Lee, and J. Park, "Design of the humanoid robot TOCABI," in *Proc. 21st Int. Conf. Humanoid Robots (Humanoids)*, Mar. 2022, pp. 322–329.

[47] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robot. Autom.*, vol. RA-3, no. 1, pp. 43–53, Feb. 1987.

[48] Y. Lee, J. Ahn, J. Lee, and J. Park, "Computationally efficient HQP-based whole-body control exploiting the operational-space formulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 5197–5202.

[49] J. Jung, S. You, and J. Park, "The operational space formulation on humanoids considering joint stiffness and bandwidth limit," *IEEE Access*, vol. 9, pp. 121883–121893, 2021.

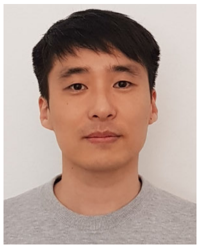
[50] J. Jung, S. Hwang, Y. Lee, J. Sim, and J. Park, "Analysis of position tracking in torque control of humanoid robots considering joint elasticity and time delay," in *Proc. 17th Int. Conf. Humanoid Robot.*, Nov. 2017, pp. 515–521.

DONGWHAN KIM (Graduate Student Member, IEEE) received the B.S. degree from the Department of Robotics, Kwangwoon University, Seoul, South Korea, in 2022. He is currently pursuing the M.S. degree with the Department of Electrical and Electronic Engineering, Korea University, Seoul. He is a Student Researcher with the Korea Institute of Science and Technology, Seoul. His research interests include robot control, manipulation, and machine learning for robot control.

SOONWOOK HWANG received the B.S. degree in mechanical and aerospace engineering from Seoul National University, South Korea, where he is currently pursuing the Ph.D. degree with the Graduate School of Convergence Science and Technology. He is a Researcher with Samsung Electronics, Hwaseong, South Korea. His research interests include whole-body control framework, multicontact control, legged locomotion, compliant control, and design and manufacturing of robots.

MYOTAEG LIM received the B.S. and M.S. degrees in electrical engineering from Korea University, Seoul, South Korea, in 1985 and 1987, respectively, and the second M.S. and Ph.D. degrees in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990 and 1994, respectively. He was a Senior Research Engineer with the Samsung Advanced Institute of Technology and an Assistant Professor with the Department of Control and Instrumentation, National Changwon University, Changwon, South Korea. Since 1996, he has been a Professor with the School of Electrical Engineering, Korea University. He has authored or coauthored more than 100 journal articles and two books: *Optimal Control of Singularly Perturbed Linear Systems and Application*: High-Accuracy Techniques, Control Engineering Series, (Marcel Dekker, 2001) and *Optimal Control: Weakly Coupled Systems and Applications*, Automation and Control Engineering Series (CRC Press, 2009). His research interests include optimal and robust control, vision-based motion control, and intelligent vehicle systems. He is a fellow of the Institute of Control, Robotics and Systems, and a member of the Korea Institute of Electrical Engineers. He serves as an Editor for *International Journal of Control, Automation, and Systems*.

YONGHWAN OH received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Pohang University of Science and Technology, Pohang, South Korea, in 1991, 1993, and 1999, respectively. From 1999 to 2000, he was a Japan Society for the Promotion of Science Postdoctoral Fellow with the Prof. A. Takanishi Laboratory, Department of Mechanical Engineering, Waseda University, Tokyo, Japan. He joined the Korea Institute of Science and Technology, Seoul, South Korea, in 2001, where he was the Head of the Center for Robotics Research, from 2011 to 2017. He is currently a Principal Research Scientist with the Korea Institute of Science and Technology. His current research interests include mechanical design of robotic systems, walking and whole-body balance control of humanoid robots, handarm coordination, and motion/interaction control of redundant manipulators.



YISOO LEE (Member, IEEE) received the B.S. and M.S. degrees in naval architecture and ocean engineering and the Ph.D. degree in intelligent convergence systems from the Department of Intelligent Convergence Systems, Seoul National University, Seoul, South Korea, in 2008, 2010, and 2017, respectively. From 2017 to 2018, he was a Postdoctoral Researcher with Seoul National University. From 2018 to 2020, he was a Postdoctoral Researcher with the Humanoids and Human Centered Mechatronics Laboratory, Istituto Italiano di Tecnologia, Genoa, Italy. He is currently a Senior Researcher with the Korea Institute of Science and Technology, Seoul. His research interests include humanoid robots, robot locomotion, manipulation, optimal control, and machine learning for robot control.