A 65-nm Humanoid Robot System-on-Chip Using Time-Domain 3-D Footstep Planning and Mixed-Signal ZMP Gait Scheduler With Inverse Kinematics

Qiankai Cao, *Graduate Student Member, IEEE*, Juin Chuen Oh, and Jie Gu, *Senior Member, IEEE*

***Abstract*— This work presents a footstep planning chip for humanoid robot. It integrates a time-domain graph search engine for high-level 3-D footstep planning and a mixed-signal zero moment point (ZMP) gait scheduler with neural inverse kinematics, enabling efficient low-level motion control. The key contributions of this work include a time-domain graph search engine for 3-D footstep planning, featuring 3-D search capabilities, D^* replanning for real-time adjustments, redundant path blocking, and efficient result readout. In addition, it introduces an energy-efficient mixed-signal ZMP gait scheduler for maintaining robot balance, along with a time-domain neural-network-based inverse kinematics module for controlling robot joints. This work is demonstrated in situ on a fully assembled robot using the 65-nm system-on-chip (SoC), achieving 2.7× energy savings for graph search and an 18.4× improvement in energy efficiency for motion control compared with prior works.**

***Index Terms*— 3-D footstep planning, humanoid robot, inverse kinematics, mixed-signal, system-on-chip (SoC), zero moment point (ZMP).**

I. Introduction

As AUTONOMOUS robotic systems have observed rapid growth in recent years, and humanoid robots are recently drawing significant interest. Compared with wheeled mobile robots, humanoid robots with human-like joint systems enable high degree-of-freedom (DOF) locomotion for complex tasks, e.g., search and rescue, housework, or medical treatment. However, there are significant challenges in motion control of such robots. First, it is complicated and computationally heavy for 3-D footstep planning on humanoid robot [1], [2] with added dimensions of height and special movements, e.g., stepping over or stepping onto objects. Most research in this area has focused on treating footstep planning as a graph search problem, commonly solved using the A^* algorithm [3], [4]. The dominant strategy involves computing an extended sequence of footsteps that lead all the way to the target, which the robot follows closely using precise motion tracking. These plans typically consist of 30 or more steps and require a few seconds of computation time. At this timescale, replanning while the robot is in motion becomes impractical. However, real-world environments frequently present unexpected events that demand immediate replanning [5]. Moving obstacles can change direction unpredictably, requiring quick responses to avoid collisions. To tackle these challenges, it is crucial to have a controller with a short and reliable response time. Second, the complex 10–20 DOFs' kinematic model for robot joint control leads to high computation workload. The kinematics problem plays a crucial role in robotic motion control. Forward kinematics refers to mapping from joint space to Cartesian task space, while inverse kinematics involves mapping from Cartesian task space to joint space [6]. Due to the complexity of inverse kinematics [6], it is typically more challenging to solve compared with forward kinematics. Moreover, when a robot manipulator performs motion control, the computational demands of inverse kinematics can significantly consume CPU resources, slowing down the robot's performance. Addressing this issue is therefore essential for improving efficiency. Third, to maintain balance, special trajectory control of the robot's center of mass (CoM) through zero moment point (ZMP) needs to be judiciously performed for fall prevention. To enable certain behaviors in humanoid robots, such as walking, motion planners must simultaneously consider the dynamic effects of the resulting motion [7]. This is because these robots rely on ground reaction forces at their supporting foot or feet, which are inherently unilateral. This constraint is effectively captured by the concept of the ZMP [8]. The unilateral nature of the contact forces translates into the requirement that the ZMP must remain within the boundaries of the supporting polygon. When this condition is satisfied, the robot avoids rotation along the edges of the supporting polygon, ensuring stable contact as long as friction remains sufficient. This is crucial for maintaining stable control of humanoid robots.

Previous research has explored 2-D path planning, including wavefront expansion of the A^* algorithm for graph searches [3], [9], the design of a path-planning processor for 2-D/3-D autonomous navigation of micro robots [10], and the development of an FPGA-based motion-planning accelerator for dual-arm robot manipulation systems [11].

Received 31 August 2024; revised 1 January 2025 and 2 February 2025; accepted 6 February 2025. Date of publication 19 February 2025; date of current version 28 March 2025. This article was approved by Associate Editor Sugako Otani. This work was supported in part by the National Science Foundation under Grant CCF-1846424. *(Corresponding author: Qiankai Cao.)*

The authors are with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: qiankaicao2019@u.northwestern.edu; juinoh2024@u.northwestern.edu; jgu@northwestern.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2025.3541484.

Digital Object Identifier 10.1109/JSSC.2025.3541484

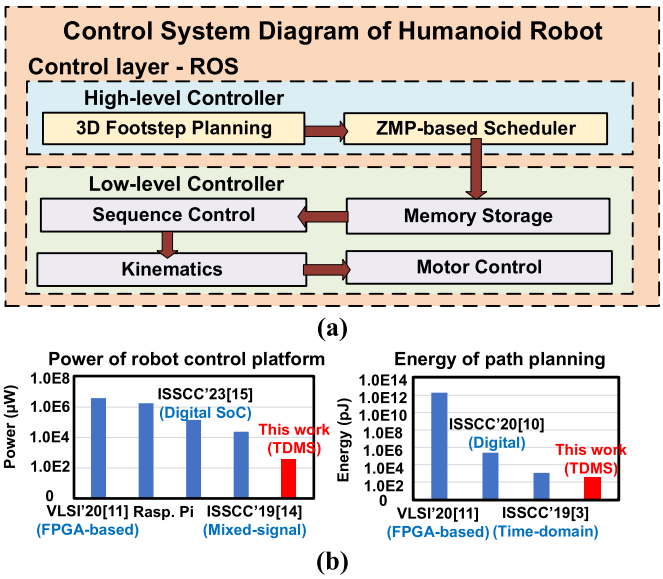


Fig. 1. (a) Humanoid robot control diagram. (b) Power of robot control platform and energy of path planning.

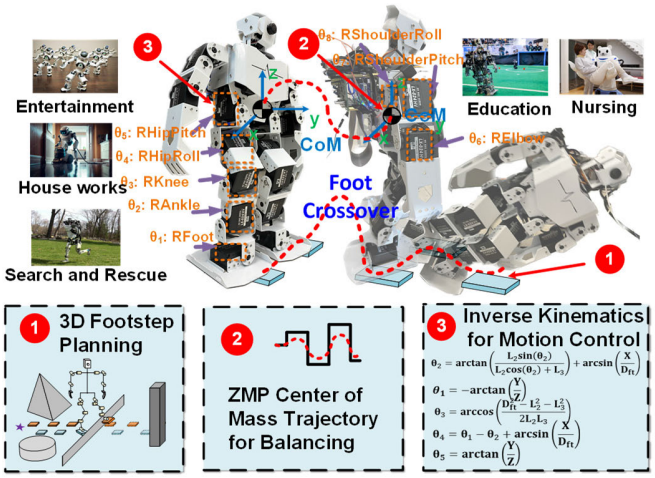


Fig. 2. Challenges of humanoid robot control.

However, these studies do not support specialized humanoid movements, such as stepping on or over obstacles. There are also mixed-signal designs for robot control, including an oscillator-based NeuroSLAM accelerator for mobile wheeled robots [12], a time-domain mixed-signal neuromorphic accelerator with reinforcement learning for autonomous micro-robots [13], and an efficient mixed-signal accelerator developed for real-time swarm intelligence [14]. These designs achieve remarkable energy efficiency for wheeled robots due to their mixed-signal nature. However, they cannot support more complex joint systems, such as those found in humanoid robots, which typically control many more motors than the 2–4-DoF systems used in wheeled robots. Some works are digital-based for robots but have not been demonstrated on real robots to showcase their full capabilities. For instance, a motion-control ASIC was designed primarily for industrial robot arms [15], a ray-casting accelerator was developed for edge robotics in augmented reality applications [16], and an artificial intelligence processor with PVT compensation was created for micro robots [17].

Despite the advancements in other works, a system-on-chip (SoC) solution specifically for humanoid robots has been lacking until now. As depicted in Fig. 1(a), a humanoid robot control system comprises a high-level controller and a low-level controller. The 3-D footstep planning module and ZMP-based scheduler collaborate to generate steps and the corresponding CoM trajectory toward the target. Once high-level plans are established, the detailed CoM trajectories are stored in memory and sent to the kinematic model with sequence control, after which joint control signals are converted into motor control signals. As shown in Fig. 1(b), compared with previous robot control platforms [3], [10], [11], [14], [15], this approach achieves a two–four orders of magnitude reduction in energy consumption for path planning, representing a significant improvement.

To address the aforementioned challenges and improve the performance of humanoid robot control, as shown in Fig. 2, this work introduces a 65-nm mixed-signal SoC designed explicitly for humanoid robots, offering several key innovations: 1) a time-domain [18], [21], [22] graph search engine for 3-D footstep planning featuring 3-D search, D^* replanning [19] for on-the-fly adjustment, blocking of redundant paths and efficient readout of search results; 2) a highly efficient mixed-signal ZMP gait scheduler, crucial for maintaining balance in humanoid robots; 3) a time-domain neural-network-based inverse kinematic module for robot joint control; and 4) in situ demonstrations on a real assembled robot with the 65-nm SoC rendering 2.7× overall energy saving for graph search and 18.4× higher energy efficiency for motion control compared with prior works.

The rest of this article is organized as follows: Section II presents an overview of the SoC top-level architecture and the robot assembly. Section III delves into the time-domain graph search ASIC for 3-D footstep planning and D^* replanning [19], including circuit details for the vertex, direction lock (DL), vertex lock and unlock module, delay cells, and scan chain tracing-back modules. Section IV covers the ZMP gait scheduler and neurokinematics for low-level motion control. Section V showcases the implementation and measurement results obtained from the test chip, along with real robot demonstrations and retrain methods with use cases. Section VI concludes this article. This article is an extended version of the conference publication in [20].

II. Chip Top-Level Architecture and Robot System

A. Chip Top-Level Architecture

Fig. 3(a) and (b) illustrates the top-level architecture of the chip, which includes: 1) a 40 × 40 time-domain graph search engine with specialized mixed-signal circuits for high-level 3-D footstep planning; 2) a ZMP gait trajectory generator for controlling the CoM for robot balancing; 3) a hybrid time-digital domain neural network serving as an inverse kinematic estimator for joint control; and 4) a motor control module with UART to manage external motors via CAN bus. Upon completion of high-level footstep planning, the ZMP gait scheduler module passes the CoM trajectory in (X, Y, Z)

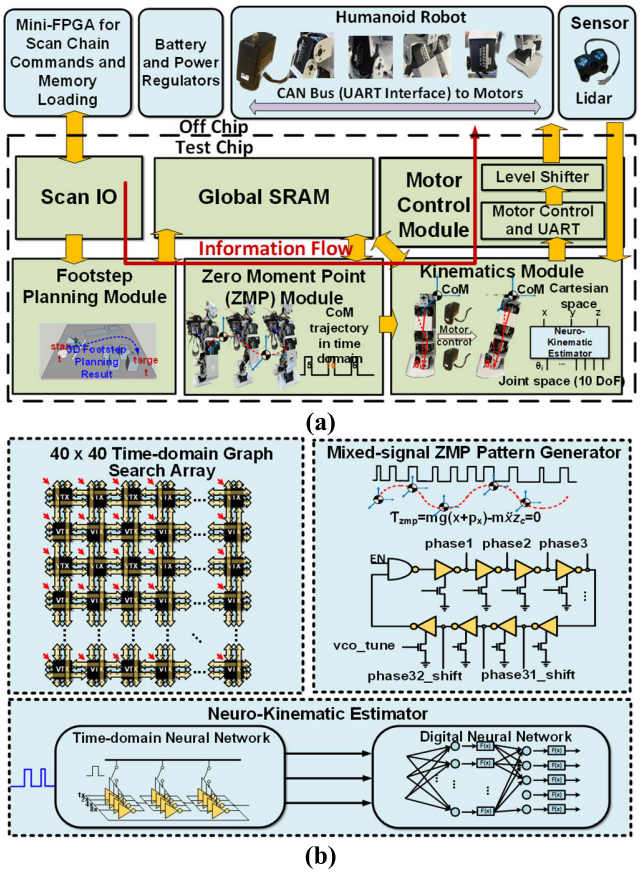


Fig. 3. (a) Top-level architecture of the developed SoC. (b) Circuits' implementation for sub-modules of the proposed SoC.

format to low-level joint control. The system then transitions to low-level control, where the neurokinematic module converts the Cartesian space of end-effectors into the 10-DoF joint space for each motor. Final motion control commands are transmitted through the motor control module using the CAN bus and UART protocol.

B. Robot Assembly and Special Movement

Fig. 4 showcases the assembled robot system [23], highlighting both the front and back views of its motor and control systems. The system includes a mini-FPGA that imports control signals and communicates with the test chip via a scan cable and driver board for direct motor control. The test chip, mounted on a demo board, transmits joint angle control signals through the UART interface using a CAN bus. In this work, the map input is processed by the test chip for 3-D footstep planning, followed by the generation of the ZMP trajectory pattern. The resulting data are then transmitted to the kinematic module for low-level motor control.

Compared with wheeled robots [13], [14], the motion controller for humanoid robots introduces several unique features tailored to their distinct movements. Instead of visualizing the environment as a 2-D grid map, the humanoid robot converts a 3-D floor map into a 3-D grid map that includes the height of objects. This allows the robot to decide whether to take shortcuts by stepping over obstacles that would typically block the way in a 2-D scenario. The ability to step on or over

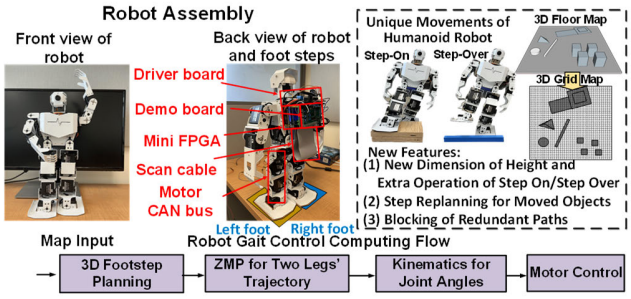


Fig. 4. Humanoid robot assembly and features with gait control computing flow.

obstacles enables the robot to navigate around them and reach its target via a shorter route. In addition, the chip supports efficient path replanning on the go when the environment changes suddenly, and it blocks redundant paths to make quick decisions with reduced power consumption. These features enhance the effectiveness and energy efficiency of humanoid robot control.

III. Time-Domain Graph ASIC for 3-D Footstep Planning and D^* Replanning

A. 3-D Footstep Planning and D^* Algorithm for Replanning

Three-dimensional footstep planning [24], [25], [26] can be modeled as a shortest path problem using the A^* algorithm [27]. In Fig. 5(a), the floor map is transformed into a grid map, where white grids represent walkable areas and darker grids indicate objects or platforms of varying heights. Some objects are marked as traversable if their height is manageable for the robot to step over or on, while others are too tall and are considered obstacles, requiring the robot to find an alternate route. For instance, a rod on the map might be low enough for the robot to step over, so a 3-D footstep planning approach would allow the robot to do so. In contrast, a 2-D path-finding method would still treat the rod as an obstacle, forcing a detour and resulting in a much longer path. After the initial 3-D footstep planning is completed, replanning is triggered when a change in the environment, such as a fallen pillar, is detected, as illustrated in Fig. 5(b). The robot follows the original 3-D footstep plan, indicated by the blue dotted arrow, until it encounters the new obstacle. At this point, a replanning algorithm is used to generate an updated path, shown by the yellow dotted arrow, which takes the environmental change into account. The robot then abandons the previous route and follows the newly generated path from the updated starting point toward the target. Unlike the widely used A^* algorithm, this work uses a more advanced D^* replanning algorithm [19], allowing the robot to adjust its path dynamically while moving toward its destination. The steps of the D^* algorithm are detailed in Fig. 6. First, mapping information is scanned into the graph ASIC for shortest path calculation. Next, the shortest path is identified by the graph ASIC, represented by the red arrows. When an environmental change occurs, the affected nodes are detected, and a global unlock signal is sent to these nodes and their successors—defined as all the vertices originating from the changed nodes. The connections to these successors are shown with blue

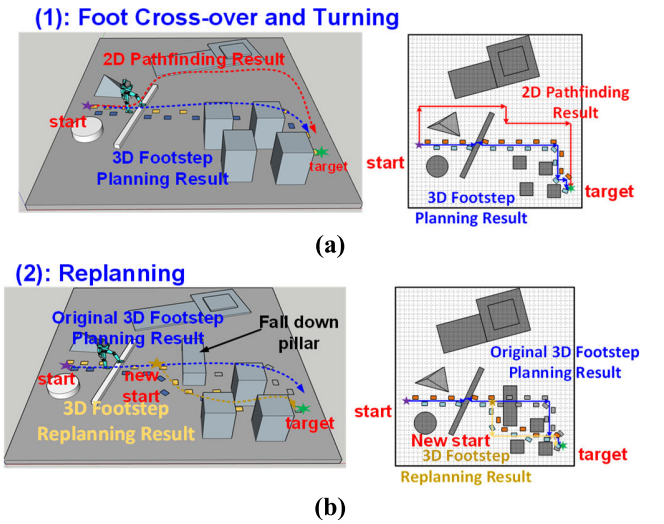


Fig. 5. (a) Comparison of 3-D footstep planning and 2-D path finding problem. (b) 3-D footstep replanning with changed environment.

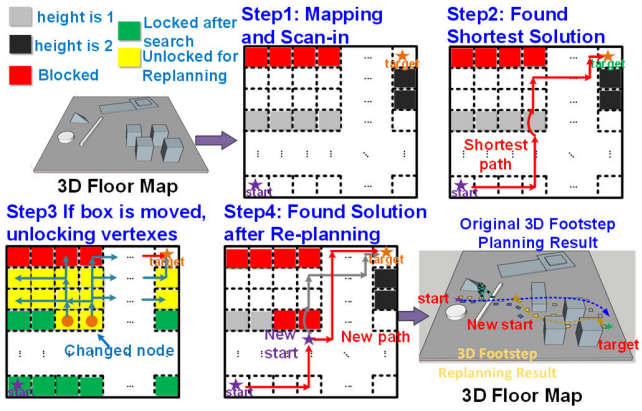


Fig. 6. Footstep replanning example of D^* algorithm.

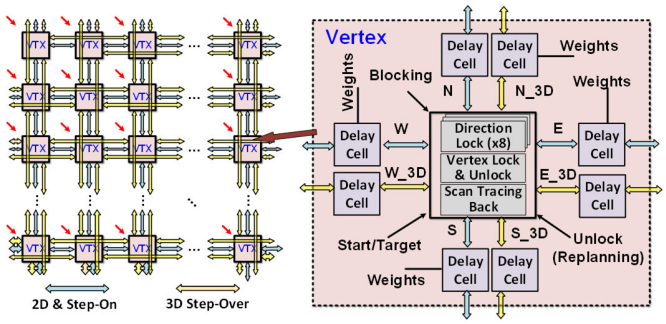


Fig. 7. Time-domain graph ASIC and vertex design.

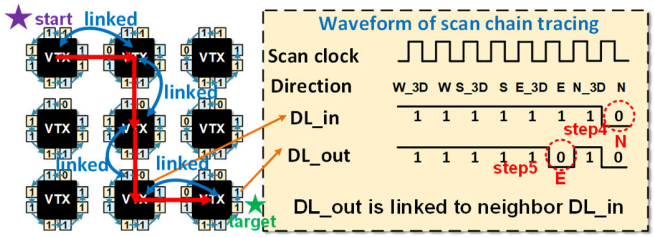


Fig. 8. Waveform of the scan chain tracing function.

arrows. Finally, a new starting point is set at the changed node, triggering another round of graph search that only updates the direction of the changed nodes and their successors. This D^* approach achieves 1.8× energy savings compared with the traditional A^* algorithm, which requires recalculating the entire map. In this example, the changed nodes are blocked, causing the new path to detour around them, following the yellow arrows to reach the target.

B. Time-Domain Graph ASIC for 3-D Footstep Planning

Fig. 7 elaborates details of the 3-D footstep planning and the time-domain graph search engine. While 2-D occupancy grid maps typically meet the needs of wheeled robots [13], humanoid robots necessitate additional terrain height information to account for special movements of stepping over/onto objects in 3-D space. Different from the widely used A^* algorithm, a more sophisticated D^* replanning algorithm [19] was adopted in this work, enabling the robot to adjust its path while heading to the destination. In the time-domain circuit implementation shown in Fig. 7, a 40 × 40 vertex array is deployed to generate locomotion trajectory. The mapping information, e.g., distance of single step and height of stairs, are mapped into a programmable 2-bit delay cell at interconnect of the vertexes. Inside each vertex, time-domain signals are passed from eight directions, including four planar directions similar as prior work [3] and another four directions for the new dimension of height for stepping-over or stepping-on movements. Each "vertex lock" circuit includes multiple NAND, NOR gates, and a DFF for catching the earliest time-domain signal and producing a "Lock" signal. A set of DL modules are used to record the direction of the first-come time-domain signal. The time-domain signals propagate as a wavefront through the whole map resulting in the shortest path being locked in the DL circuits.

Besides static planning, this work also supports D^* on-the-fly replanning when the environment is changed, e.g., an object moved by the robot. Finally, rather than a full memory scan outputting all the direction values as in [3], this design enables only tracing back along the shortest path using the direction information. As shown in Fig. 8, the shortest path is calculated by the graph search engine and is indicated by a red arrow. During the calculation, the vertices along the path are linked based on the DL results. In the waveform, "0" signifies the earliest arriving direction. If multiple "0"s appear, it means the earliest arriving signals come from multiple directions, but only one direction is selected as the result. For example, if the target vertex's DL_in shows a "0" in the "E" direction, the vertex in the "E" direction will be linked to the target vertex. The DL_out will then be scanned into the target vertex after all the DL_in bits are sequentially scanned out, following the sequence from steps 5 to 4, step 3, and so on, all the way back to the start vertex.

As shown in Fig. 9, thanks to the described technique, this work achieves a 29.1× speedup compared with the full memory scan approach used in previous research [3]. The replanning process delivers an average of 1.8× energy savings compared with methods that do not use replanning. Overall, the study realizes a 2.7× reduction in energy consumption

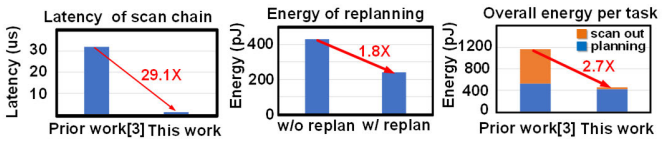


Fig. 9. Comparison to prior work on latency of scan and energy of replanning.

for path planning compared with prior work [3], due to the low-power techniques implemented in this study.

C. Circuit Design Detail of Graph Vertex

As illustrated in Fig. 10(a), the delay cell encodes mapping information, such as single-step distances and stair heights, into a programmable 2-bit delay cell located at the vertex interconnections. In the figure, "00" denotes a standard single step, "01" represents a step-on, "10" signifies a step-over, and "11" is used for debugging purposes. The delay of each cell reflects the time cost of its corresponding configuration. An analog signal, Vtune, globally adjusts the delay across all the cells. Each vertex receives time-domain signals from eight directions. In Fig. 10(b) and (c), a set of DL modules record the direction of the first-arriving signal. For example, in the DL (W) module, if W is not the earliest signal, a "trig" pulse sets DL_W to "1" to indicate the direction. If W is the earliest signal, no "trig" pulse is generated, leaving DL_W at "0." Each "vertex lock" circuit includes multiple NAND and NOR gates, plus a D flip-flop (DFF) to capture the earliest signal and generate a "Lock" signal for further processing. When the direction "S" is the earliest arriving signal, the DFF clock signal toggles, producing the "Lock" signal simultaneously.

In addition to static planning, this work supports D^* on-the-fly replanning for dynamic environments. Sometimes, the environment could change due to robot behavior, such as a robot kicking or moving an object with robot hands. As a result, the 3-D footstep planning needs to be updated based on the latest map information. If redoing the entire map in A star algorithm, it will be unnecessary, as the robot has already moved to the new start point. So, a D star algorithm enables only partially redoing 3-D footstep planning and efficiently finding the shortest path using updated map information. In hardware implementation, the unlock function is managed by the circuit shown in the bottom right in Fig. 10. For the changed nodes, an unlock_global signal is issued to reset the vertex lock DFF. For their successors, the unlock signal propagates along with the DL information to identify all the nodes that need to be reset. For blocking functions, a predefined configuration will be set for those nodes that are at the corner of the map. It will stop the vertices from propagating the pulses to avoid unnecessary path planning. This approach achieves an average energy saving of 32.9% across 50 random map search tasks.

IV. ZMP Gait Scheduler and Neurokinematics for Low-Level Motion Control

A. Cart-Table Model and ZMP Gait Scheduler

Following our discussion on high-level 3-D footstep planning, let us move on to low-level gait scheduler and

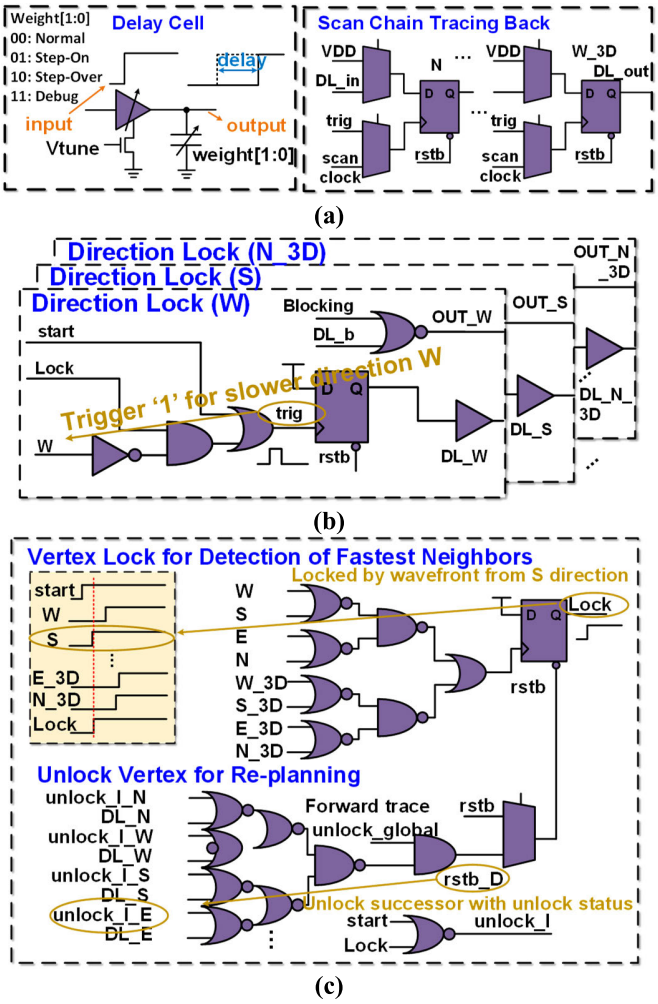


Fig. 10. Circuit design details of a graph ASIC vertex. (a) Delay cell and scan chain tracing-back module, (b) DL module of the vertex, and (c) vertex lock module.

neurokinematic for motion control. Unlike robot cars or other types of multi-legged robots, a humanoid robot must consider maintaining balance. To address this, we developed a ZMP-based gait scheduler to maintain stability while walking. As illustrated in Fig. 11, the dynamics of a walking robot can be described using a cart-table model. In the figure, to counteract the momentum generated by the car's gravity (CoM) "mg," the cart must move with an acceleration of $\ddot{x}$. And the torque τ_{ZMP} at the supporting p_x is shown as this formula. It is forced to be zero to control the robot to be stable. By considering the ZMP dynamics, we can design a gait scheduler to ensure the robot's CoM follows a specific trajectory, preventing the robot from falling. Fig. 12 provides details on the ZMP-based CoM control for the gait scheduler and the neurokinematic circuits for robot joint control. The ZMP is the location where the total moment of the robot at the ground is zero. For dynamic stability, the ZMP must remain within the support region of the robot. ZMP is used to establish the target trajectory for the robot's CoM. This mixed-signal ZMP phase generator achieves 3.4× power savings compared with an equivalent digital solution.

A mixed-signal circuit [35], [36], [37] featuring a VCO and multiplexers (MUXs) is used to generate a sinusoidal CoM

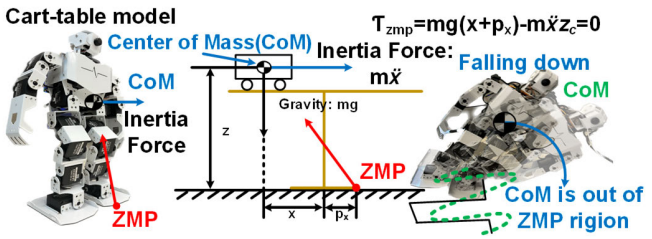


Fig. 11. Robot cart-table model of ZMP.

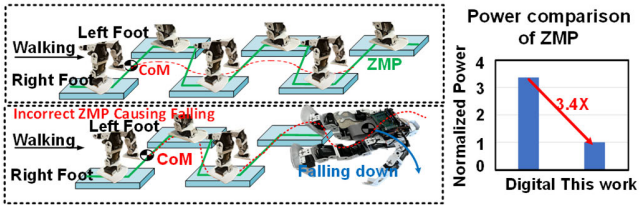


Fig. 12. Comparison of ZMP trajectory for balance and power.

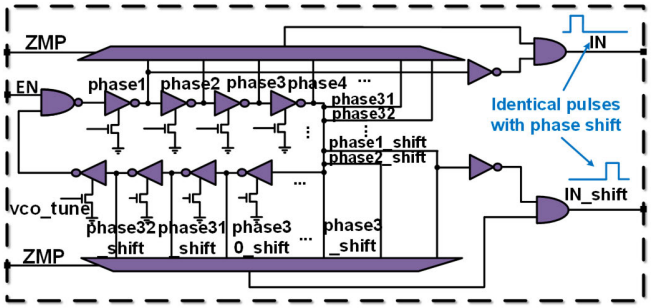


Fig. 13. Mixed-signal ZMP phase generator.

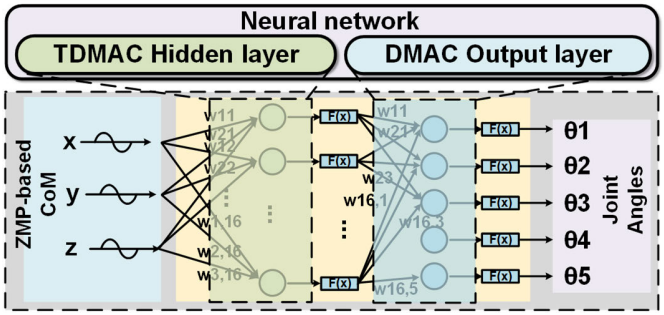


Fig. 14. Neural estimator for inverse kinematics.

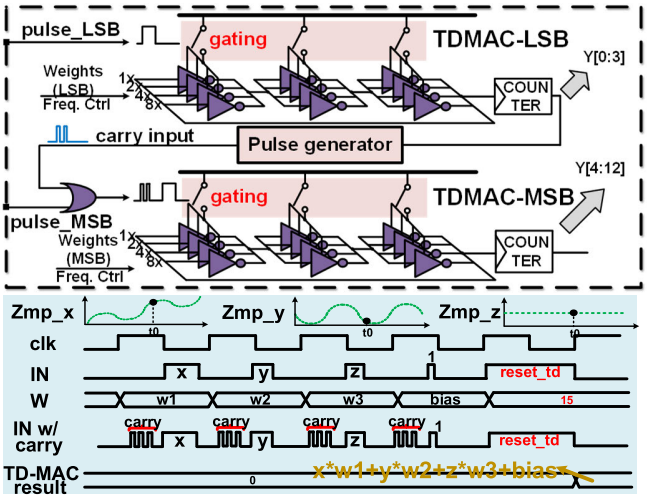


Fig. 15. Circuit and waveform of TDMAC with bit partition.

trajectory that ensures dynamic stability, commonly used for simple pattern generation. In practical humanoid gait planning, more flexible CoM trajectories, such as polynomial-based ones, are also widely adopted. In this work, a sinusoidal CoM trajectory (X, Y, Z) is produced and encoded as time pulses for inverse kinematics computation. As shown in Fig. 13, a ring oscillator with linear increased phases is used to convert digital sine wave-like CoM trajectory, x, y, z into time pulses. In comparison, digital solutions typically rely on LUTs, which require many flip-flops for implementation [38]. In this work, digital information is encoded as time delays, reducing power consumption by avoiding the use of DFFs. In addition, the phases are split into ZMP_LSB and ZMP_MSB for TDMAC operation. This time-domain interface ensures compatibility with the subsequent neurokinematic module.

B. Neurokinematics for Inverse Kinematics

As shown in Fig. 14, due to the highly complex trigonometric computation in IK, a neural network is used to approximate the calculation. A neural network consisting of a TDMAC as the hidden layer and a DMAC as the output layer is used to convert CoM trajectory (x, y, z) in Cartesian space into θ_i in joint space, a process known as inverse kinematics. This calculation is particularly complex when the DoFs in robot systems are high. Fig. 15 details the hardware implementation of neurokinematics, featuring an 8-bit time-domain MAC [18], [22], [28] developed using a bit partition technique. In this design, a pulse from the ZMP module controls the propagation duration of the ring oscillator, while digital weights regulate the propagation speed of the ring. The result is captured by a counter, effectively performing an MAC operation in the time domain. To further enhance energy efficiency, the 8-bit weight is divided into LSB and MSB, with a pulse generator acting as a time-domain carry signal. When the LSB result counter is full, it triggers the pulse generator to create a carry pulse, which then controls the gating signal along with the ZMP input MSB.

As shown in Fig. 16, the proposed neural kinematics module demonstrates a 2% loss compared with the ideal inverse kinematic model, with an additional 1% accuracy loss due to time-domain implementation. This is acceptable, as both the ZMP method can tolerate errors within the ZMP region, and the neural network is resilient to errors introduced by the proposed mixed-signal scheme. This method achieves a 7.5× area saving using the time-domain MAC with bit partition technique, along with a 1.8× latency reduction and a 12.1× area saving for inverse kinematics compared with its digital counterpart.

C. Neural Estimator With Retrain Methodology

Fig. 17 shows a two- to three-layer fully connected neural network trained to map ZMP-based CoM sine wave-interpolated trajectories to joint angles, using a training set

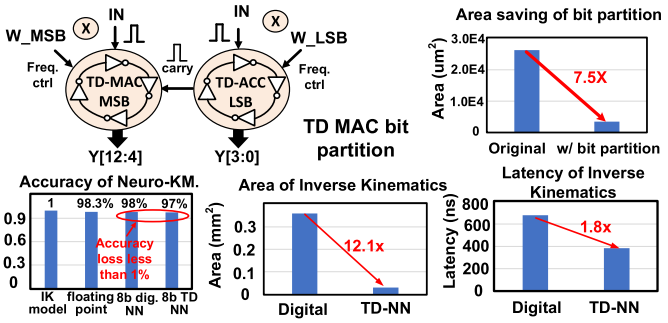


Fig. 16. Comparison of area and power with bit partition.

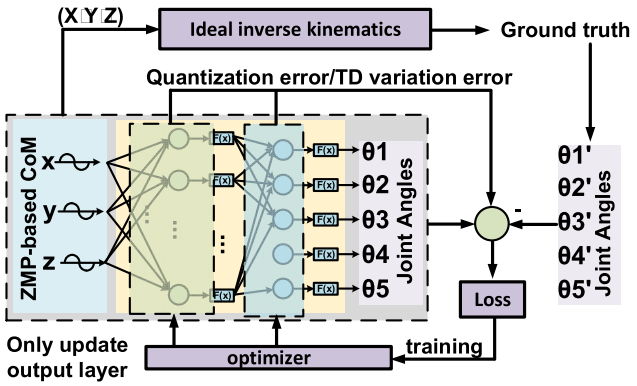


Fig. 17. Block diagram of neural estimator retrain.

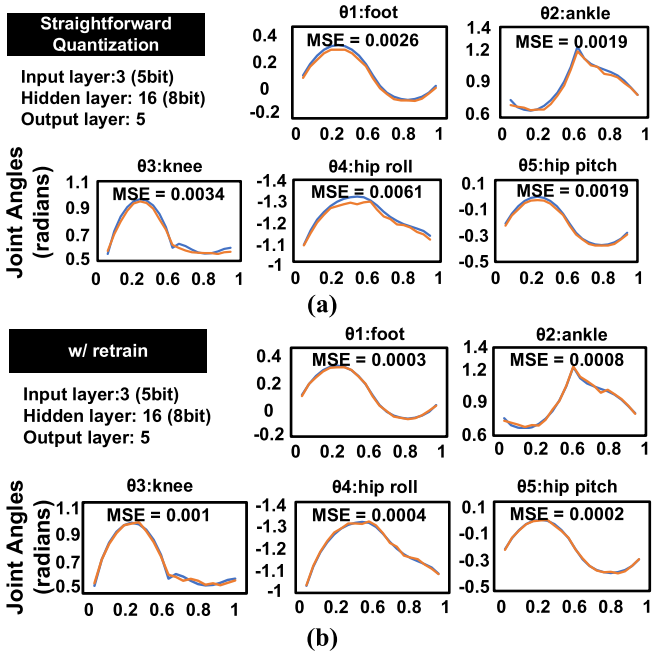


Fig. 18. MSE of neural estimator output with quantization and retrain methodology. (a) MSE of neural estimator output with quantization. (b) MSE of neural estimator after retraining.

derived from an ideal inverse kinematic model with functions such as arctan, arcsin, and arccos. A sine-wave-shaped CoM trajectory, generated under ZMP constraints, is used to compute joint angles (θ_1 foot, θ_2 ankle, θ_3 knee, θ_4 hip roll, and θ_5 hip pitch) via the ideal inverse kinematic model, shown as blue curves. This model serves as a reference for the neural network, with training aimed at minimizing the MSE between their outputs using ground-truth data as labels. Initially, the model undergoes the first round of training. However, errors due to quantization or time-domain variations arise, as shown in Fig. 18(a). To address these errors, a second round of retraining is conducted, updating only the output layer, which significantly reduces the model's MSE, as depicted in Fig. 18(b). After optimization, the model is deployed on the robot to evaluate its performance. Including real CoM trajectories in training prevents unexpected Cartesian-to-joint space mappings, ensuring all possible trajectories are sampled to avoid model failure.

V. Measurement Results and Evaluation for Test Case

A. Chip Implementation

A mixed-signal 3-D footstep planning SoC was designed using a 65-nm CMOS process. The real-world demonstration setup, shown in Fig. 19, includes the test chip and environment. Fig. 20 presents the chip micrograph and implementation details. The active die area measures 3.34 mm^2 (2 $\times$ 1.67 mm), with a nominal supply voltage of 1.0 V, a maximum frequency of 1 MHz, and peak power consumption of 432.8 μW at 1.2 V. The graph search engine features 40 $\times$ 40 vertices, 1600 $\times$ 8 edges, and a node delay of 1.1 ns at 1.2 V.

B. Measurement Results

Fig. 21 presents the measurement results for CoM trajectory, linearity, vertex delay, and power breakdown. In the ZMP-based CoM trajectory measurement, the designed ZMP trajectory is encoded as time-domain delays at time frames t2 and t5 in the first row, with the corresponding measured robot swing distances for the actual CoM trace shown in the second, third, and fourth rows. The integral non-linearity (INL) of the TDMAC is less than 1 LSB. For vertex delay, the search rate reaches 910M operations at 1.2 V, with peak power consumption of 432.8 μW. The power breakdown indicates 207.9 μW consumption for the graph search array and 185.2 μW for digital circuits, including memory banks and DMAC. The power consumption for the ZMP/VCO and TD NN circuit is also shown at 1 V running at 1 MHz.

Fig. 22 presents 2-D view and wavefront result from the chip. The top row shows the results of normal 3-D footstep planning with blocking. In the 3-D view, the robot steps over obstacles and walks toward the target, following the planned steps. The blue line in the 2-D view represents the path projected onto the grid map. In the wavefront result, brighter colors indicate longer delay, and pulses propagate across the entire map as a wavefront, except for blocked nodes.

C. Comparison to Prior Works

Table I illustrates a comparison table with prior works. In the comparison table, this work presents the first 3-D footstep planning SoC chips for humanoid robots. It supports both high-level footstep planning and low-level motion control. In terms of energy efficiency for control, this work

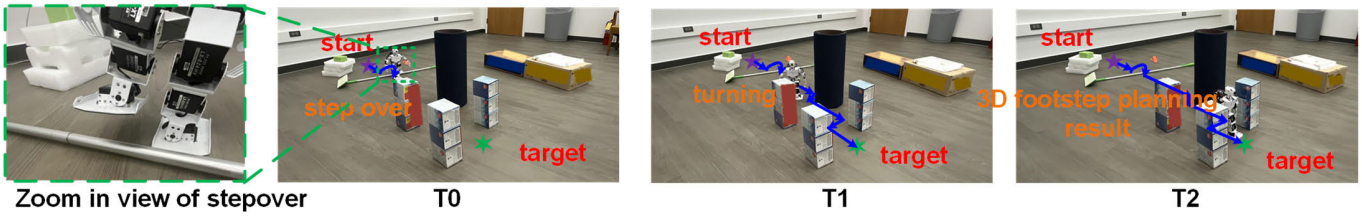
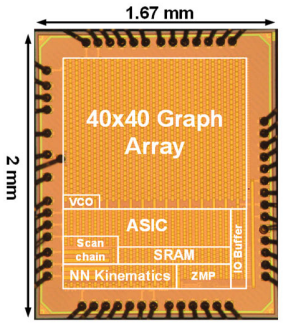


Fig. 19. Robot demonstration across multiple time frames.



Applications	3D footstep planning + motion control
Technology	65nm LP CMOS
Architecture	Mixed-signal ASIC
# of vertices	40x40
# of edges	1600x8
Voltage	1V
Frequency	1MHz
Peakpower	432.8uW @1.2V
Delay per node	1.1ns @1.2V

Fig. 20. Chip micrograph and specifications.

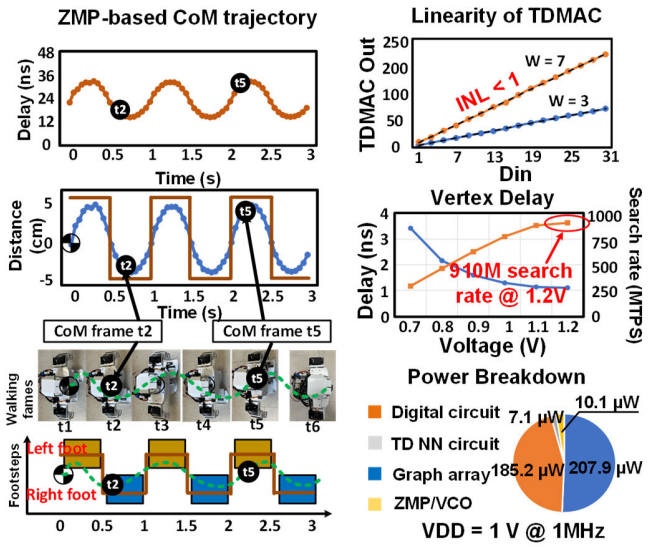


Fig. 21. Measured ZMP-based CoM trajectory, linearity, vertex delay, and power breakdown.

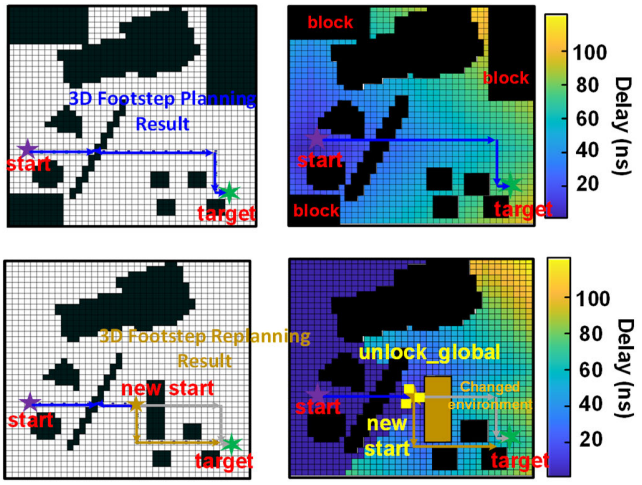


Fig. 22. 3-D footstep planning and wavefront result.

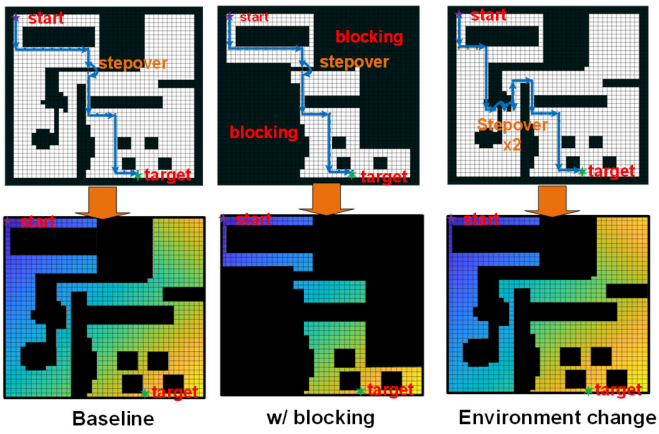


Fig. 23. Footstep planning with blocking and replanning.

achieves 645 Hz/mW, which is 18.4× higher than previous work, thanks to the mixed-signal circuit implementations. The energy efficiency for the neural network ranges from 3.2 to 6.5 TOPS/W. For path planning, this work demonstrates a more complex 3-D footstep planning with a 1.6× higher search rate and an overall 2.7× improvement in energy per task due to low-power features.

D. Demonstration and Evaluation of Test Cases

Fig. 23 illustrates more complex graph scenarios involving blocking and replanning algorithms. In the baseline scenario, the robot can step over a narrow rod and make several turns to reach the target. In the blocking example, two corners of the graph are obstructed, preventing the time-domain signal from propagating through the blocked vertices. Consequently, the footstep planning [29], [30] remains unchanged from the baseline. In the environmentally changing scenario, the narrow rod becomes a wider obstacle, preventing the robot from stepping over it. As a result, the robot detours from its original path and finds a new route to reach the target.

Fig. 24 shows the test board setup, which includes the test chip and LDOs to supply the chip's voltage source. An XTAL is used to generate a precise clock for producing motor control signals via the UART protocol. A level shifter converts the motor control signal from the on-chip 1.8 to 3.3 V for motor control. In the test environment, a 3 × 3 m area with a 40 × 40 grid map is constructed, featuring several paper boxes as obstacles [31], [32], [33], [34]. A mop is placed on the floor, enabling the robot to step over the rod and find a shorter path compared with a wheeled robot. To further enhance the scalability of the proposed methods, the graph engine can

TABLE I
COMPARISON TABLE WITH PRIOR WORKS

		ISSCC'19 [3]	ISSCC'20 [12]	ISSCC'19 [14]	ISSCC'23 [15]	This Work
Hardware		65nm/Time	Mixed-signal	65nm	28nm/Digital	65nm/MS Time
Total Area (mm2)		0.4	5	2	3.56	3.34
Application		A* shortest path	SLAM	Swarm intelligence	Motion control	3D footstep plan + Motion control
Graph Size		40 x 40	7x7	-	-	40 x 40
Memory Size		2.3 kB	37.9 kB	16kB	-	22kB
Frequency		-	78.2–130.8MHz	1kHz-1.5MHz	200MHz	1MHz
Peak Power		26.4mW	17.87mW	3.4uW	142mW	432.8uW @1.2V
Energy Efficiency	for control	-	-	-	35 Hz/mW	654 Hz/mW[1)]
	for NN	-	8.0 TOPS/W	1.1-9.1 TOPS/W	-	3.2-6.5 TOPS/W
Path Planning	Energy per task	1166.2 pJ[2)]	-	-	-	424.7 pJ
	Search rate (edge/sec)	559M[3)]	-	-	-	910M[3)] @1.2V

1) Efficiency for control = Control Rate/Power (20Hz control rate is used in this work from motor spec.)
2) a 55% cache access energy is used as reported in [4] 3) MTEPS=Million Traversed Edges Per Second

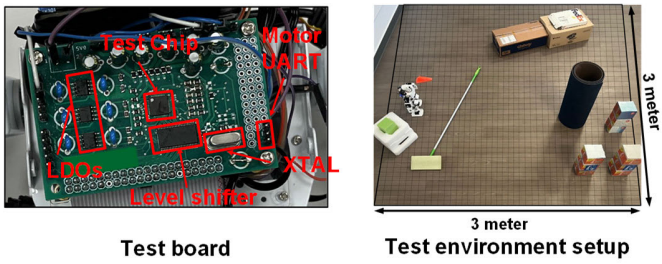


Fig. 24. Test board and robot test environment setup.

be adapted as a path planning engine for various types of robots, including quadrupeds and robotic cars. In addition, the low-level neurokinematic module is highly scalable and can be applied to any jointed robot, such as quadrupeds, which also require real-time inverse kinematic calculations. This flexibility makes the proposed system versatile and applicable to a wide range of robotic platforms, paving the way for broader adoption across diverse robotic applications.

VI. CONCLUSION

This work presents a 65-nm SoC chip for humanoid robot control with in situ demonstration. It features a time-domain graph search engine for 3-D footstep planning. D^* on-the-fly replanning and a tracing back method along the shortest path with blocking of redundant paths are proposed. A mixed-signal ZMP-based gait scheduler is developed to maintain robot balance, and a mixed-signal neurokinematic module is developed for inverse kinematics in motion control. Overall, this approach achieves 2.7× energy saving in graph search and an 18.4× improvement in energy efficiency for motion control compared with previous works.

REFERENCES

[1] J. Garimort, A. Hornung, and M. Bennewitz, "Humanoid navigation with dynamic footstep plans," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 3982–3987.

[2] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the honda ASIMO humanoid," in *Proc. IEEE Int. Conf. Robot. Autom.*, Barcelona, Spain, Apr. 2005, pp. 629–634.

[3] L. R. Everson, S. S. Sapatnekar, and C. H. Kim, "A 40×40 four-neighbor time-based in-memory computing graph ASIC chip featuring wavefront expansion and 2D gradient control," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2019, pp. 50–52.

[4] Y. Zhou, X. Jin, and T. Wang, "FPGA implementation of a algorithm for real-time path planning," *Int. J. Reconfigurable Comput.*, vol. 2020, Aug. 2020, Art. no. 8896386.

[5] M. Fujita, "AIx robotics: Technology challenges and opportunities in sensors, actuators, and integrated circuits," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2019, pp. 276–278.

[6] J. Wang, S. Liu, B. Zhang, and C. Yu, "Inverse kinematics-based motion planning for dual-arm robot with orientation constraints," *Int. J. Adv. Robotic Syst.*, vol. 16, no. 2, Mar. 2019, Art. no. 1729881419836858.

[7] S. Kajita et al., "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, Taipei, Taiwan, Sep. 2003, pp. 1620–1626, doi: 10.1109/ROBOT.2003.1241826.

[8] J. Park and Y. Youm, "General ZMP preview control for bipedal walking," in *Proc. IEEE Int. Conf. Robot. Autom.*, Rome, Italy, Apr. 2007, pp. 2682–2687, doi: 10.1109/ROBOT.2007.363870.

[9] C. Yu, Y. Su, J. Lee, K. Chai, and B. Kim, "A 32×32 time-domain wavefront computing accelerator for path planning and scientific simulations," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Austin, TX, USA, Apr. 2021, pp. 1–2.

[10] C. Chung and C.-H. Yang, "A 1.5μJ/task path-planning processor for 2D/3D autonomous navigation of micro robots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2020, pp. 324–326.

[11] A. Kosuge and T. Oshima, "A 1200×1200 8-edges/vertex FPGA-based motion-planning accelerator for dual-arm-robot manipulation systems," in *Proc. IEEE Symp. VLSI Circuits*, Honolulu, HI, USA, Jun. 2020, pp. 1–2.

[12] J.-H. Yoon and A. Raychowdhury, "A 65 nm 8.79TOPS/W 23.82 mW mixed-signal oscillator-based NeuroSLAM accelerator for applications in edge robotics," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2020, pp. 478–480.

[13] A. Amravati, S. B. Nasir, S. Thangadurai, I. Yoon, and A. Raychowdhury, "A 55 nm time-domain mixed-signal neuromorphic accelerator with stochastic synapses and embedded reinforcement learning for autonomous micro-robots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers* , San Francisco, CA, USA, Feb. 2018, pp. 124–126.

[14] N. Cao, M. Chang, and A. Raychowdhury, "A 65 nm 1.1-to-9.1TOPS/W hybrid-digital-mixed-signal computing platform for accelerating model-based and model-free swarm robotics," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2019, pp. 222–224.

[15] I.-T. Lin et al., "A 28 nm 142 mW motion-control SoC for autonomous mobile robots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2023, pp. 1–3.

[16] M. Kar et al., "A ray-casting accelerator in 10 nm CMOS for efficient 3D scene reconstruction in edge robotics and augmented reality applications," in *Proc. IEEE Symp. VLSI Circuits*, Honolulu, HI, USA, Jun. 2020, pp. 1–2.

[17] Y. Kim, D. Shin, J. Lee, Y. Lee, and H.-J. Yoo, "A 0.55 V 1.1 mW artificial-intelligence processor with PVT compensation for micro robots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Jan. 2016, pp. 258–259.

[18] Y. Toyama, K. Yoshioka, K. Ban, S. Maya, A. Sai, and K. Onizuka, "An 8 bit 12.4 TOPS/W phase-domain MAC circuit for energy-constrained deep learning accelerators," *IEEE J. Solid-State Circuits*, vol. 54, no. 10, pp. 2730–2742, Oct. 2019.

[19] S. Koenig and M. Likhachev, "D*lite," in *Proc. 18th Nat. Conf. Artif. Intell.*, 2002, pp. 476–483.

[20] Q. Cao, J. Chuen Oh, and J. Gu, "A mixed-signal 3D footstep planning SoC for motion control of humanoid robots with embedded zero-moment-point based gait scheduler and neural inverse kinematics," in *Proc. IEEE Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Honolulu, HI, USA, Jun. 2024, pp. 1–2.

[21] Q. Cao, X. Chen, and J. Gu, "Development of tropical algebraic accelerator with energy efficient time-domain computing for combinatorial optimization and machine learning," in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design (ISLPED)*, Vienna, Austria, Aug. 2023, pp. 1–6.

[22] Z. Chen, S. Fu, Q. Cao, and J. Gu, "A mixed-signal time-domain generative adversarial network accelerator with efficient subthreshold time multiplier and mixed-signal on-chip training for low power edge devices," in *Proc. IEEE Symp. VLSI Circuits*, Honolulu, HI, USA, Jun. 2020, pp. 1–2.

[23] V. Honkote et al., "A distributed autonomous and collaborative multi-robot system featuring a low-power robot SoC in 22 nm CMOS for integrated battery-powered minibots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, 2019, pp. 48–50.

[24] M. Missura and M. Bennewitz, "Fast footstep planning with aborting a," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 2964–2970.

[25] K. Harada, S. Hattori, H. Hirukawa, M. Morisawa, S. Kajita, and E. Yoshida, "Motion planning for walking pattern generation of humanoid," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2007, pp. 4227–4233.

[26] A.-C. Hildebrandt, D. Wahrmann, R. Wittmann, D. Rixen, and T. Buschmann, "Real-time pattern generation among obstacles for biped robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, Sep. 2015, pp. 2780–2786.

[27] M. Psotka, F. Duchoň, M. Roman, T. Michal, and D. Michal, "Global path planning method based on a modification of the wavefront algorithm for ground mobile robots," *Robotics*, vol. 12, no. 1, p. 25, Feb. 2023.

[28] A. Sayal, S. Fathima, S. S. T. Nibhanupudi, and J. P. Kulkarni, "All-digital time-domain CNN engine using bidirectional memory delay lines for energy-efficient edge computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2019, pp. 228–230.

[29] A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz, "Anytime search-based footstep planning with suboptimality bounds," in *Proc. 12th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Osaka, Japan, Nov. 2012, pp. 674–679.

[30] D. Maier, C. Lutz, and M. Bennewitz, "Integrated perception, mapping, and footstep planning for humanoid navigation among 3D obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, Nov. 2013, pp. 2658–2664.

[31] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Madrid, Spain, Nov. 2014, pp. 279–286.

[32] F. Causa and G. Fasano, "Adaptive cooperative navigation strategies for complex environments," in *Proc. IEEE/ION Position, Location Navigat. Symp. (PLANS)*, Karlsruhe, Germany, Apr. 2020, pp. 100–111.

[33] W. Huang, C.-M. Chew, Y. Zheng, and G.-S. Hong, "Pattern generation for bipedal walking on slopes and stairs," in *Proc. 8th IEEE-RAS Int. Conf. Humanoid Robots*, Daejeon, South Korea, Dec. 2008, pp. 205–210, doi: 10.1109/ICHR.2008.4755946.

[34] M. Lapeyre, P. Rouanet, and P.-Y. Oudeyer, "The poppy humanoid robot: Leg design for biped locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, Jun. 2013, pp. 349–356, doi: 10.1109/IROS.2013.6696375.

[35] Z. Chen and J. Gu, "A time-domain computing accelerated image recognition processor with efficient time encoding and non-linear logic operation," *IEEE J. Solid-State Circuits*, vol. 54, no. 11, pp. 3226–3237, Nov. 2019.

[36] Z. Chen and J. Gu, "A scalable pipelined time-domain DTW engine for time-series classification using multibit time flip-flops with 140 Giga-cell-updates/s throughput," in *Proc. IEEE Int. Solid- State Circuits Conf. (ISSCC)*, San Francisco, CA, USA, Feb. 2019, pp. 324–326.

[37] Z. Chen, H. Zhou, and J. Gu, "Digital compatible synthesis, placement and implementation of mixed-signal time-domain computing," in *Proc. 56th ACM/IEEE Design Autom. Conf. (DAC)*, Las Vegas, NV, USA, Jun. 2019, pp. 1–6.

[38] M. Bohrn, L. Fujcik, and R. Vrba, "Novel on-chip sine wave generator," in *Proc. 34th Int. Conf. Telecommun. Signal Process. (TSP)*, Budapest, Hungary, Aug. 2011, pp. 505–508, doi: 10.1109/TSP.2011.6043679.

Qiankai Cao (Graduate Student Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2018, and the M.S. degree from Northwestern University, Evanston, IL, USA, in 2019, where he is currently pursuing the Ph.D. degree in computer engineering.

His current research interests include mixed-signal ICs and time-domain signal processing for AI and robotics.

Juin Chuen Oh received the B.S. degree in electrical engineering and the M.S. degree in computer engineering from Northwestern University, Evanston, IL, USA, in 2024.

He is currently working as an ASIC Verification Engineer at Nvidia, Santa Clara, CA, USA. His academic interests involve mixed-signal ICs and high-level synthesis.

Jie Gu (Senior Member, IEEE) received the B.S. degree from Tsinghua University, Beijing, China, the M.S. degree from Texas A&M University, College Station, TX, USA, and the Ph.D. degree from the University of Minnesota, Minneapolis, MN, USA.

He worked as an IC Design Engineer at Texas Instruments, Dallas, TX, USA, from 2008 to 2010, focusing on ultralow-voltage mobile processor design and integrated power management techniques. He was a Senior Staff Engineer in Maxlinear Inc., Carlsbad, CA, USA, from 2011 to 2014, focusing on low-power mixed-signal broadband SoC design. He is currently an Associate Professor at Northwestern University, Evanston, IL, USA. His research interests include advanced ML accelerator design, integrated power and clock management, and novel analog mixed-signal computing circuit and system.

Dr. Gu received the NSF Career Award. He has served as a Program Committees and Conference Organizer for numerous conferences such as ISSCC, CICC, ISPLED, DAC, ICCAD, ICCD, and GLSVLSI, and as an Associate Editor for JSSC, TVLSI, and TCAS-II.