IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

# Enhancing Safety and Manipulability of Redundant Manipulators: Accelerated Motion Generation in Dynamic Environments

Zongwu Xie, Mengfei Li, Wandong Sun, *Student Member, IEEE*, Baoshi Cao, Yang Liu, Zhengpu Wang, Yiming Ji, Hong Liu, *Senior Member, IEEE*, Boyu Ma, *Graduate Student Member, IEEE,* and Zhihong Wu

***Abstract*—Motion generation in dynamic environments is crucial for human-machine interaction with redundant manipulators. In this context, we propose the Enhancing Safety and Manipulability (ESM) scheme, which integrates geometry-based dynamic obstacle avoidance, manipulability optimization, trajectory tracking, and joint limit avoidance into a unified scheme operating at the joint-angle level. The introduction of a flexible collision library enables the scheme to locate critical points based on geometry, while the incorporated obstacle speed allows the scheme to effectively avoid dynamic obstacles. In the ESM, manipulability is naturally set as the non-convex goal. To solve the ESM, the Accelerated Multi-agent recurrent Neural Network (AMNN) is proposed, which uses a meta-heuristic approach to construct activation functions, endowing the neural network with non-convex control capabilities. Subsequently, a GPU-based parallel computing method is implemented, significantly reducing computing time. Detailed simulations, experiments, and comparisons demonstrate the framework's effectiveness and superiority.**

***Index Terms*—Redundant manipulator, obstacle avoidance, manipulability optimization, dynamic environments.**

## I. Introduction

IN RECENT years, manipulators have reshaped modern automation [1], playing a crucial role in industry [2], services [3], space exploration [4], and other various fields [5, 6]. Redundant manipulators possess more degrees of freedom (DOF) than required for performing specific tasks [7]. The redundant DOF grants the manipulator the capability to accomplish its primary task while considering other subtasks, such as singularity handling [8], obstacle avoidance [9], and joint physical limits avoidance [10].

### A. Relevant Literature

Kinematic tracking is a fundamental issue in manipulation tasks. However, singularities may be encountered. While facing singularity, the joint velocities and accelerations may suddenly increase, causing damage to the manipulator. To describe the proximity of a manipulator's configuration to singular configurations, Yoshikawa [11] first defined the concept of manipulability. In the follow-up studies, researchers typically maximize manipulability and avoid singularities by employing the pseudo-inverse-type formulation. Jin et al. [12] transformed the manipulability optimization problem at the joint-velocity level into a convex quadratic programming (CQP) problem and designed a generalized recurrent neural network for solving. Zhang et al. [13] also optimized manipulability at the joint-velocity level while introducing a new term to prevent system failure due to the singularities. Lu et al. [14] designed an algorithm with manipulability optimization and the control efforts in the joint-acceleration level then transformed into a CQP problem. Yang et al. [15] investigated the manipulability optimization of redundant manipulators in the joint-acceleration level, ensuring the initial velocity is zero. Ren et al. [16] innovatively proposed Riemannian geometry-based manipulability ellipsoid tracking, which provides a new idea for manipulability optimization. However, most schemes rely on continuous matrix inversion, which leads to high computational costs and poor control stability near singular points. Jin et al. [17] proposed a method to optimize manipulability in the joint-velocity level without inverting matrices for the first time, making progress in real-time manipulability optimization. However, even if matrix inversion is not required, extending non-convex functions to convex functions changes the optimal solution and may lose feasibility when dealing with multi-level joint constraints.

Another crucial goal in the manipulation tasks of redundant manipulators is obstacle avoidance. In the optimization perspective, obstacle avoidance metrics are typically formulated as equality or inequality constraints, then formulated into a quadratic programming (QP) form and solved using neural networks. Zhang et al. [18] proposed a minimum velocity norm scheme, implementing obstacle avoidance with a dynamically updated inequality constraint. However, due to the utilization of the symbolic function, the manipulator's joint velocity will change suddenly when avoiding obstacles. Therefore, Guo et al. [19] proposed a minimum acceleration-norm obstacle avoidance scheme. However, the minimum distance is used only for collision detection and is not strictly limited to avoiding obstacles, the obstacle avoidance function is at risk of failure. Chen et al. [20] proposed a minimum jerk norm scheme, minimizing jerk while implementing obstacle avoidance. However, imposing constraints at the joint-jerk level degrades performance and does not guarantee that the final joint velocity and acceleration will be zero. Ma et al. [21]

This work was supported in part by the National Natural Science Foundation of China under the Basic Science Center Program for Space Robot Intelligent Manipulation under Grant T2388101 and in part by the Special Foundation of China Postdoctoral Science under Grant 2022T150161. *(Corresponding authors: Yang Liu.)*

Zongwu Xie, Mengfei Li, Wandong Sun, Baoshi Cao, Yang Liu, Zhengpu Wang, Yiming Ji, Hong Liu, and Boyu Ma are with the State Key Laboratory of Robotics and Systems, Harbin Institute of Technology, Harbin 150080, China. (e-mail: xiezongwu@hit.edu.cn; 21b908050@stu.hit.edu.cn; 24b908020@stu.hit.edu.cn; cbs@hit.edu.cn; liuyanghit@hit.edu.cn; 21b908032@stu.hit.edu.cn; jiyiming@alu.hit.edu.cn; hong.liu@hit.edu.cn; boyu.ma@stu.hit.edu.cn).

Zhihong Wu is with the Institute of Spacecraft System Engineering, Beijing 100080, China. (e-mail: zhihong_wu@126.com).

proposed an actual shape-based obstacle avoidance synthesized by velocity-acceleration minimization scheme, making significant progress in avoiding convex obstacles, sudden changes in velocity and acceleration and the nonzero final joint velocity and acceleration. Tan et al. [22] combined obstacle avoidance and manipulability optimization at joint-angle level, but since the geometry of the manipulator and obstacles are not considered, high-precision collision avoidance could not be guaranteed. However, all these solutions do not work with dynamic obstacles, and strict safety distance limits cannot be guaranteed. Xu et al. [23] proposed a dynamic obstacle avoidance approach based on the minimum velocity norm, using an escaping velocity for handling dynamic obstacles. However, since a series of critical points are selected for each obstacle, and increase with the number of links, the computational burden becomes the primary challenge, especially when solving non-convex optimization problems.

To accelerate the solution process of the optimization algorithm, one possible solution is to implement accelerated computation using neural networks. In view of the parallel processing characteristics of neural networks, the main acceleration methods include distributing computing across multiple processors [22, 24] or GPU [25, 26]. With the gradual improvement of GPU computing power and the improvement of general interfaces, the use of GPU for parallel computing of neural networks has become mainstream.

### B. Contribution and Structure of this Article

To address the issues aforementioned, we introduce the Enhancement of Safety and Manipulability (ESM) scheme and Accelerated Multi-agent recurrent Neural Network (AMNN) for redundant manipulators' motion generation in dynamic environments. The ESM scheme integrates geometry-based dynamic obstacle avoidance, trajectory tracking, manipulability optimization, and joint physical limit avoidance into a unified scheme. By incorporating the Flexible collision library (FCL) [27], the critical points are checked with high precision. The joint-velocity level reconstruction of obstacle avoidance inequality and the introduction of obstacle speed enable ESM to perform collision-free motion generation in dynamic environments. Moreover, the manipulability in ESM is set as the optimization goal naturally without inverting matrices. The non-convex representation of the ESM preserved the original optimal solution but selecting an appropriate control method is crucial. Therefore, the multi-agent recurrent neural network is proposed for solving this problem. The AMNN constructs the activation function in a meta-heuristic approach [28, 29], allowing it to continuously approach the optimal solution in each sampling time of non-convex control. Through the update of the feasible region, the controller has the ability to process time-series trajectories. In addition, to address the high computational complexity of AMNN, a GPU-based parallel acceleration method was introduced, which significantly reduced the computation time, making progress in real-time non-convex control of redundant manipulators.

The primary contributions are as follows:

1) For the first time, geometry-based dynamic obstacle avoidance, manipulability optimization, trajectory tracking, and joint physical limits avoidance are integrated into a unified framework operating at the joint-angle level.
2) The joint-velocity level reconstruction of obstacle avoidance inequality and the introduction of obstacle speed enable collision-free motion generation in dynamic environments.
3) The AMNN incorporating metaheuristics and designed for time-varying constrained non-convex control is proposed and utilized for solving the ESM scheme.
4) A GPU-based parallel acceleration method was incorporated into AMNN and the compute time is greatly shortened.

## II. Problem Formulation And Scheme Design

In this section, we discuss the objectives and constraints of manipulator tasks. Subsequently, the ESM scheme is proposed and reformulated to the joint-angle level.

### A. Kinematic Model

The forward kinematics of redundant manipulators [30] can be written as

$$\boldsymbol{r}(t) = f\left(\boldsymbol{\theta}(t)\right), \tag{1}$$

where $\boldsymbol{r}(t) \in R^m$, $\boldsymbol{\theta}(t) \in R^n$, $t$ is the discrete timestep, and $n > m$. $f(\cdot)$ is the nonlinear transformation from joint angles to the end-effector. To solve the nonlinearity, the derivative of (1) is taken with respect to the time and (1) is converted into the velocity level:

$$\dot{\boldsymbol{r}}(t) = \boldsymbol{J}\left(\boldsymbol{\theta}(t)\right)\dot{\boldsymbol{\theta}}(t), \tag{2}$$

where $\boldsymbol{J}\left(\boldsymbol{\theta}(t)\right) \in R^{m\times n}$ represents the Jacobian matrix from the joint space to the Cartesian space.

### B. Manipulability

Manipulability is a measure of the ease with which a manipulator can move and exert forces in different directions from a given position and orientation. According to [11], the manipulability is given by

$$\mu\left(\boldsymbol{\theta}(t)\right) = \sqrt{\det\left(\boldsymbol{J}\left(\boldsymbol{\theta}(t)\right)\boldsymbol{J}\left(\boldsymbol{\theta}(t)\right)^T\right)} = \mu_1\mu_2\cdots\mu_m, \tag{3}$$

where $\mu_i \in R$ is the eigenvalue of $\boldsymbol{J}\boldsymbol{J}^T$.When the eigenvalue corresponding to a principal axis of the ellipsoid approaches zero, the manipulator reaches the singularity configuration, where the manipulator loses at least one DOF and cannot move in at least one direction.

### C. Obstacle Avoidance in Dynamic Environments

For engineering applications, it is essential for manipulators to effectively avoid obstacles in dynamic environments. The first measure involves determining the critical point $\boldsymbol{M}_l$ and $\boldsymbol{N}_k$, which is defined as the nearest point on link $L_l$ and obstacle $O_k$. We utilize a tool that is called FCL [27] with geometry-based collision detection capabilities, it accepts the convex hull as input and outputs the closest point. Critical points are calculated as

$$\left(\boldsymbol{M}_l, \boldsymbol{N}_k\right) = FCL\left(conv\left(L_l\right), conv\left(O_k\right)\right). \tag{4}$$

The critical point $\boldsymbol{M}$ and $\boldsymbol{N}$ across the manipulator and obstacles is given by

$$(\boldsymbol{M},\boldsymbol{N}) = \arg\min\left(D(\boldsymbol{M}_l,\boldsymbol{N}_k)\right), \tag{5}$$

where $D(\cdot)$ is the distance function. The obstacle avoidance issue can be formalized as $\|\boldsymbol{M}-\boldsymbol{N}\|_2 \geq d$, where $d$ represents the minimum safe distance as a threshold. The velocity level reconstruction [23] is given by

$$\frac{\partial\|\boldsymbol{M}-\boldsymbol{N}\|_2}{\partial t} \geq -Y\left(\frac{1}{1+\exp(-|D|)}-\frac{1}{2}\right)\mathrm{sgn}(D), \tag{6}$$

With

$$D = \|\boldsymbol{M}-\boldsymbol{N}\|_2 - d,$$

where $Y>0$ is used to adjust the velocity limits, $\mathrm{sgn}(\cdot)$ is the sign function. The left-side of (6) can be formulated as

$$\begin{aligned}\frac{\partial\|\boldsymbol{M}-\boldsymbol{N}\|_2}{\partial t} &= \frac{\partial\sqrt{(\boldsymbol{M}-\boldsymbol{N})^T(\boldsymbol{M}-\boldsymbol{N})}}{\partial t}\\ &= \frac{1}{\|\boldsymbol{M}-\boldsymbol{N}\|_2}(\boldsymbol{M}-\boldsymbol{N})^T(\dot{\boldsymbol{M}}-\dot{\boldsymbol{N}})\\ &= \boldsymbol{u}_{\boldsymbol{NM}}{}^T(\dot{\boldsymbol{M}}-\dot{\boldsymbol{N}}),\end{aligned} \tag{7}$$

where $\boldsymbol{u}_{\boldsymbol{NM}}{}^T \in R^3$ is the unit directional vector from point $\boldsymbol{M}$ to $\boldsymbol{N}$. Assume $\boldsymbol{\zeta}$ is the unit vector of the rotation axis of link $L_l$, $\boldsymbol{p}_i$ represents the position of the joint $i$, Jacobian matrix of $\boldsymbol{M}$ can be calculated [30] as

$$\begin{aligned}\boldsymbol{J}_i &= \begin{bmatrix}\boldsymbol{\zeta}_i \times \boldsymbol{p}_i\\ \boldsymbol{\zeta}_i\end{bmatrix},\\ \boldsymbol{J}_M &= [\boldsymbol{J}_1,\ \boldsymbol{J}_2,\ \cdots,\ \boldsymbol{J}_n].\end{aligned} \tag{8}$$

And $\dot{\boldsymbol{M}}$ can be given by (2). Hence, (6) can be formulated as

$$\boldsymbol{\delta}_k\dot{\boldsymbol{\theta}}(t) \leq YV - V_0, \tag{9}$$

With

$$\begin{aligned}\boldsymbol{\delta}_k &= -\boldsymbol{u}_{\boldsymbol{NM}}{}^T\boldsymbol{J}_M(\boldsymbol{\theta}(t)),\\ V_0 &= \boldsymbol{u}_{\boldsymbol{NM}}{}^T\dot{\boldsymbol{N}},\\ V &= \left(\frac{1}{1+\exp(-|D|)}-\frac{1}{2}\right)\mathrm{sgn}(D).\end{aligned}$$

Through (9), the approach speed of the manipulator and dynamic obstacle is limited, ensuring that the closest distance at the next sampling time is greater than $d$.

### D. Scheme Design and Reformulation

The ESM scheme synergizes geometry-based dynamic obstacle avoidance, manipulability optimization, trajectory tracking, and joint physical limits avoidance given by

$$\begin{aligned}\min_{\boldsymbol{\theta}(t)}\quad & \boldsymbol{F}(\boldsymbol{\theta}) = h_1\boldsymbol{e}(\boldsymbol{\theta}) + h_2\frac{1}{\mu(\boldsymbol{\theta})}\\ s.t.\quad & \boldsymbol{\theta}^- \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}^+\\ & \dot{\boldsymbol{\theta}}^- \leq \dot{\boldsymbol{\theta}} \leq \dot{\boldsymbol{\theta}}^+\\ & \boldsymbol{\delta}_k\dot{\boldsymbol{\theta}} \leq YV - V_0\end{aligned}, \tag{10}$$

where $\boldsymbol{e}(\boldsymbol{\theta}) = \|\boldsymbol{r}(\boldsymbol{\theta}(t)) - \boldsymbol{r}_d\|_2$ is the tracking error, $h_1$ and $h_2$ are used to adjust the weights of position tracking and manipulability, $\boldsymbol{r}_d \in R^3$ represents the tracking target. Assume that the joint velocity and obstacle velocity are continuous at the sampling time, the forward difference formula is employed as

$$\begin{aligned}\min_{\boldsymbol{\theta}(t)}\quad & \boldsymbol{F}(\boldsymbol{\theta}) = h_1\boldsymbol{e}(\boldsymbol{\theta}) + h_2\frac{1}{\mu(\boldsymbol{\theta})}\\ s.t.\quad & \boldsymbol{\theta}^- \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}^+\\ & \boldsymbol{\delta}_k\left(\frac{\boldsymbol{\theta}(t)-\boldsymbol{\theta}(t-\Delta t)}{\Delta t}\right) \leq YV - V_0\end{aligned}, \tag{11}$$

With [21]

$$\begin{aligned}\boldsymbol{\theta}^+ &= \min\left(\Delta t\dot{\boldsymbol{\theta}}^+ + \boldsymbol{\theta}(t-\Delta t), \boldsymbol{\theta}^+\right),\\ \boldsymbol{\theta}^- &= \max\left(\Delta t\dot{\boldsymbol{\theta}}^- + \boldsymbol{\theta}(t-\Delta t), \boldsymbol{\theta}^-\right).\end{aligned}$$

where $\Delta t$ represents the sampling time.

*Remark 1*: In ESM, trajectory tracking is set as an optimization objective rather than an equality constraint, which allows natural task slacking with hard constraints without additional work.

*Remark 2*: (11) is not suitable for trajectory control at the initial sampling time. For the starting configuration, it can be given directly from an external source, or generated by modifying (11), removing the velocity component in (11), and adjusting to $D \geq 0$.

## III. Control System Design

In this section, the AMNN is proposed for time-varying constrained non-convex control. Subsequently, the convergence analysis and a GPU acceleration are presented for AMNN.

### A. Multi-Agent Recurrent Neural Network

The inspiration for the AMNN comes from artificial rabbit optimization (ARO) [31]. Inspired by ARO, the MNN is constructed for non-convex control with time-varying constraints.

The agent neuron in AMNN is constructed in a meta-heuristic approach, it oscillates between collaborative adaptation and incremental adjustments. The shift is driven by the decreasing energy factor

$$A_i(l) = 4\left(1-\frac{l}{L}\right)\ln\frac{1}{r_1}, \tag{12}$$

where $i = 1,\cdots,P$, $P$ represents the size of agent neurons in $l-th$ layer, $l$ is the current layer, $L$ is the number of all layers, $A_i(l)$ represents the energy factor of the agent $i$ in $l-th$ layer, and $r_1$ is a random number within (0, 1). The agent neuron is in the collaborative adaptation phase when $A_i(l) > 1$, and in the incremental adjustments phase when $A_i(l) < 1$.

For collaborative adaptation, joints randomly selected to undergo mutation can be described as

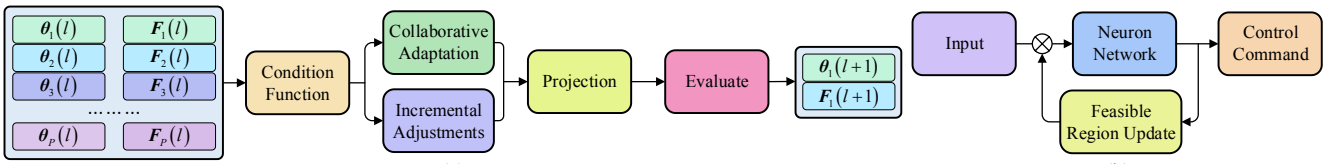


Fig. 1 Schematic of the AMNN. (a) The metaheuristic activation function of the agent neuron. (b) The control strategy diagram.

$$\boldsymbol{v}_{ca}=\begin{cases}1 & if\ k_1==\boldsymbol{r}(g)\\ 0 & else\end{cases}, \tag{13}$$

where $k_1=1,2,\cdots,n$ and $g=1,\cdots,\lceil r_2 n\rceil$. $r_2$ is a random number within (0, 1), $n$ represents the dimension of the agents' DOF, $\lceil\cdot\rceil$ is a ceiling function, $\boldsymbol{r}$ is a randomly arranged integer array from 1 to $n$, $\boldsymbol{r}(g)$ returns a random integer from 1 to $n$, and $\boldsymbol{v}_{ca}\in R^n$. The learning operator used to describe the mutation length are

$$\boldsymbol{H}_i(l)=\left(\exp(1)-\exp\left(\frac{l-1}{L}\right)^2\right)\sin(2\pi r_3)\boldsymbol{v}_{ca}, \tag{14}$$

where $\boldsymbol{H}\in R^n$, $r_3$ is a random number within (0, 1). The candidate output through collaborative adaptation is calculated as

$$\boldsymbol{p}_i(l+1)=\boldsymbol{\theta}_i(l)+\boldsymbol{H}\circ(\boldsymbol{\theta}_j(l)-\boldsymbol{\theta}_i(l)), \tag{15}$$

where $j$ is a random integer from 1 to $P$, $i\neq j$, the symbol $\circ$ represents the Hadamard Product.

For incremental adjustments, a specific joint angle is selected to undergo mutation. It can be defined as follows

$$\boldsymbol{v}_{ia}=\begin{cases}1 & if\ k_2==\lceil r_4 n\rceil\\ 0 & else\end{cases}, \tag{16}$$

where $k_2=1,2,\cdots,n$, $r_4$ is a random number within (0, 1), and $\boldsymbol{v}_{ia}\in R^n$. Randomly generated configurations $\boldsymbol{n}_i(l)$ near $\boldsymbol{\theta}_i(l)$ for updating is given by

$$\boldsymbol{n}_i(l)=\boldsymbol{\theta}_i(l)+\left(\frac{L-l+1}{L}r_5\right)\boldsymbol{v}_{ia}\circ\boldsymbol{\theta}_i(l), \tag{17}$$

where $r_5$ is a random number within (0,1).

As the iteration processes, the distance between $\boldsymbol{n}_i$ and $\boldsymbol{\theta}_i(l)$ continually decreases. The candidate angles through incremental adjustments can be calculated as

$$\boldsymbol{p}_i(l+1)=\boldsymbol{\theta}_i(l)+\boldsymbol{H}\circ(r_6\boldsymbol{n}_i-\boldsymbol{\theta}_i(l)), \tag{18}$$

where $r_6$ is a random number within (0, 1).

To ensure that the adjusted joint angles of the agent remain within the feasible region, the projection can be described as

$$\begin{aligned}\boldsymbol{p}_{\Omega b}(l+1)&=\max(\boldsymbol{\theta}^-,\min(\boldsymbol{p}_i(l+1),\boldsymbol{\theta}^+)),\\ \boldsymbol{p}_{\Omega i}(l+1)&=\begin{cases}\boldsymbol{p}_{\Omega b}(l+1) & if\ (9)\\ \boldsymbol{\theta}_i(l) & if\ not\ (9)\end{cases}.\end{aligned} \tag{19}$$

After the projection, the output is calculated as

$$\begin{aligned}\boldsymbol{F}_i(l+1)&=\begin{cases}\boldsymbol{F}_i(l) & \boldsymbol{F}_i(l)\le\boldsymbol{F}(\boldsymbol{p}_{\Omega i}(l+1))\\ \boldsymbol{F}(\boldsymbol{p}_{\Omega i}(l+1)) & \boldsymbol{F}_i(l)>\boldsymbol{F}(\boldsymbol{p}_{\Omega i}(l+1))\end{cases},\\ \boldsymbol{\theta}_i(l+1)&=\begin{cases}\boldsymbol{\theta}_i(l) & \boldsymbol{F}_i(l)\le\boldsymbol{F}(\boldsymbol{p}_{\Omega i}(l+1))\\ \boldsymbol{p}_{\Omega i}(l+1) & \boldsymbol{F}_i(l)>\boldsymbol{F}(\boldsymbol{p}_{\Omega i}(l+1))\end{cases},\end{aligned} \tag{20}$$

where $\boldsymbol{F}(\cdot)$ is the objective function. Through continuous metaheuristic calculation for any agent neuron in each layer, at the output layer, the optimal joint angle for the ESM can be calculated as

$$\boldsymbol{\theta}(t)=\arg\max(\boldsymbol{F}_{1\ldots P}(L)). \tag{21}$$

Through a loop solution, the sequence of manipulator joint angles can be obtained.

Fig. 1 displays the nonlinear activation function of the agent neuron and the control strategy of the AMNN. We consider each individual in the AMNN as an independent neuron, and the iteration of the algorithm as the flow from layers to layers. The AMNN comprises $L+1$ layers with each layer except the last containing $P$ agent neurons. The first layer performs initialization at the beginning of the loop. Each neuron receives joint angles $\boldsymbol{\theta}_{1\ldots P}(l)$ and fitness $\boldsymbol{F}_{1\ldots P}(l)$ from the previous layer, and outputs the $\boldsymbol{\theta}_i(l+1)$ and $\boldsymbol{F}_i(l+1)$ of the corresponding agent.

### *B. Theoretical Analysis*

Since the update of agent neurons is meta-heuristic, the convergence process of the AMNN can be described using a Markov model.

*Theorem 1:* The update process of agent neurons of the AMNN in layers is a finite homogeneous Markov chain.

*Proof:* According to Fig. 1(a), the neurons in the current layer receive a finite number of joint positions $\boldsymbol{\theta}_{1\ldots P}(l)$ and fitness $\boldsymbol{F}_{1\ldots P}(l)$ from the previous layer. According to (20), the joint angle $\boldsymbol{\theta}_{1\ldots P}(l+1)$ and the fitness $\boldsymbol{F}_{1\ldots P}(l+1)$ output by the agent neurons in the current layer, which is the information received by the next layer of agent neurons, are determined by $\boldsymbol{F}_i(l)$ and $\boldsymbol{F}_i(\boldsymbol{p}_{\Omega i}(l+1))$. According to (19), $\boldsymbol{p}_{\Omega i}(l+1)$ is related to $\boldsymbol{p}_i(l+1)$, $\boldsymbol{\theta}^-$ and $\boldsymbol{\theta}^+$. According to (15) and (18), $\boldsymbol{p}_i(l+1)$ is related to the joint angle value $\boldsymbol{\theta}_i(l)$, the learning operator $\boldsymbol{H}$, and other coefficients. Among them, $\boldsymbol{\theta}_i(l)$ is only related to the agent neurons in the previous layer. Thus, the update process of agent neurons is a finite homogeneous Markov chain.

Moreover, the convergence of AMNN can be judged using

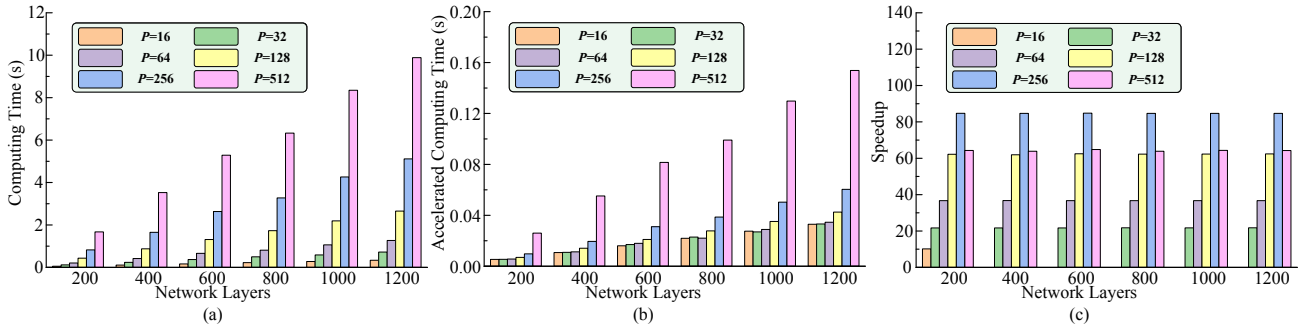


Fig. 2 Comparison of computing time. (a) Computing time using CPU-based serial computation. (b) Computing time using GPU-based parallel computation. (c) Multiple of acceleration

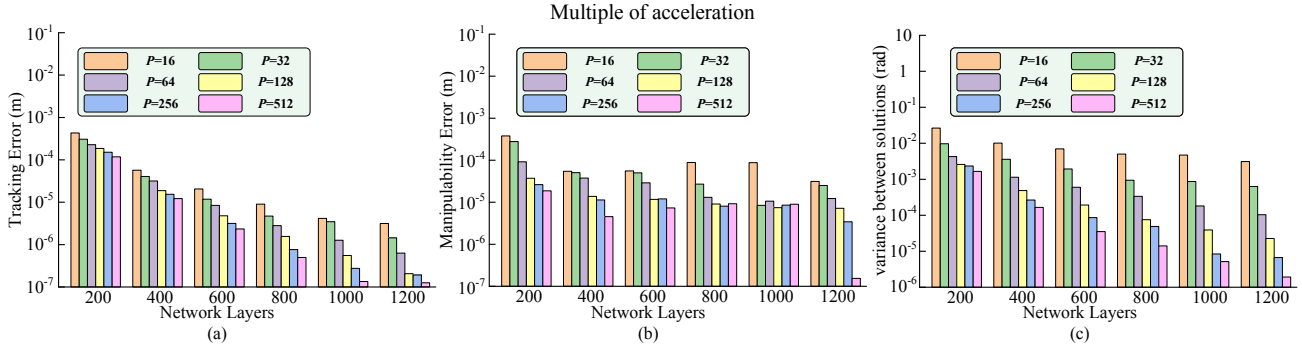


Fig. 3 Results analysis with different parameters. (a) tracking error $e_T$ . (b) Manipulability error $e_M$ . (c) Variance between the solutions $e_\theta$

the convergence criterion of stochastic search algorithms.

*Theorem 2:* The AMNN can converge to the global optimal solution in each control loop.

*Proof:* The probability that AMNN fails to find the global optimal solution in an infinite layer network approaches 0 [31]. As $L$ gradually increases, we can obtain

$$\prod_{L=0}^{\infty}\left[1-\mu_L\left(\boldsymbol{A}\right)\right]=0, \qquad \lim_{L\to\infty} P\left(\boldsymbol{\theta}_L \in \boldsymbol{R}_{\varepsilon,M}\right)=1. \tag{22}$$

where $\mu_L\left(\boldsymbol{A}\right)$ is the probability that $\boldsymbol{A}$ satisfies the global convergence criterion of heuristic algorithms, $\boldsymbol{R}_{\varepsilon,M}$ represents the optimality region. Since AMNN is a meta-heuristic process that satisfies the global search convergence theorem [32], it can be inferred that AMNN can converge to the global best solution at each sampling time.

### C. Acceleration with GPU

For offline tasks, simply increasing the number of network layers will bring better results, however, the real-time performance and the closeness to the optimal solution need to be carefully weighed due to the complex metaheuristic activation functions of AMNN.

[25] propose a GPU acceleration method of meta-heuristic optimization using Compute Unified Device Architecture (CUDA). Following [25], AMNN uses GPU to accelerate the function evaluation and parallel computing of neurons. Its core is to divide the neurons in the same layer into $B$ blocks and compute in GPU. Each block contains $T$ threads

$$T=\min\left(\frac{1024}{P_b}, 2^{\lceil \log_2 n\rceil}\right), \tag{23}$$

where $P_b$ represents the number of agent neurons in a block. The acceleration performance is further presented below.

Based on the above analysis, it can be seen that the main difference between ARO and AMNN lies in the fact that we significantly shorten the solution time of the algorithm by reconfiguring ARO into a neural network-like representation, where each individual is considered as an independent neuron, and algorithmic iterations are considered as a flow between layers, and parallel computation is carried out using GPUs. This approach significantly speeds up the solution of non-convex optimization algorithms and allows their application to real-time tracking and obstacle avoidance problems for realistic manipulator.

## IV. Discussion and Simulation

In this section, we discuss the effectiveness of GPU acceleration, and a suitable set of parameters is selected for non-convex control. Subsequently, simulations are performed to verify the effectiveness of the proposed ESM and AMNN.

### A. Discussion on Parameters

We first verify the performance of the acceleration. The experimental setup consists of a system with an NVIDIA RTX 4090 and an Intel i7-14700KF. We use the Franka Emika Panda manipulator for simulation and the joint physical limitations follow the official information [33]. Robot simulation is performed in NVIDIA Isaac Sim. The experiments were designed to compare the computation time using CPU-based serial computation and GPU-based parallel computation. The experimental method is as follows: for the fixed desired position of the end effector, we perform C=200 calculations each and take the average computation time considering 200, 400, 600, 800,1000, 1200 of network layers and 16, 32, 64, 128, 256, 512 neurons per layer. The desired position is set as $\boldsymbol{T}=\left[0.6,0.3,0.5\right]^T$ m. The hyperparameters in ESM are set as $h_1=1$, $h_2=1$, $Y=200$. Fig. 2(a) shows the computation time

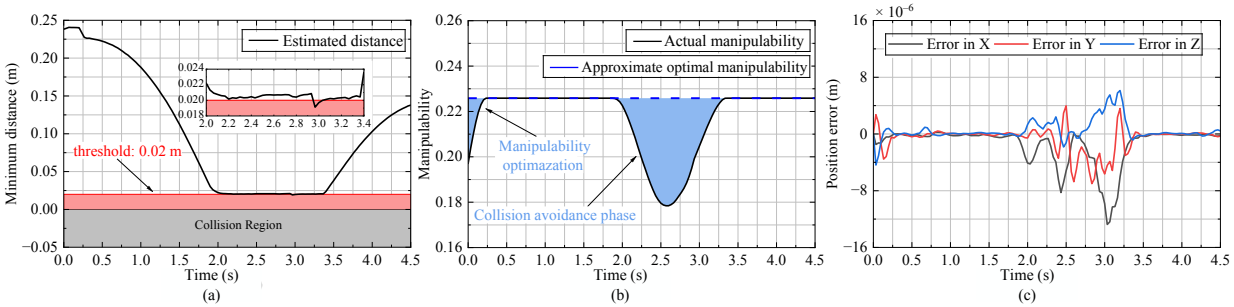


Fig. 4 Results of manipulability optimization in dynamic environments. (a) The closest distance between the manipulator and the obstacle. (b) Manipulability. (c) Tracking errors.

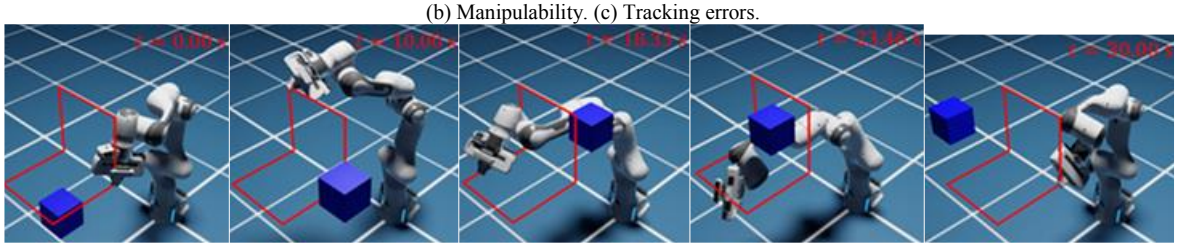

Fig. 5 Trajectory tracking in dynamic environments, where the obstacle velocity direction changes every 5s.

with CPU-based serial computation and Fig. 2(b) and Fig. 2(c) show that the computation time is greatly reduced with the acceleration, and the speedup is up to 80 times. It is noted that the computation time of parallel computing does not increase linearly like serial computation and, the speedup depends on the number of neurons in each layer but not the number of network layers. This is because in AMNN, the computation between neurons in the same layer is performed in parallel, while the layer-to-layer update is a finite homogeneous Markov chain, which is computed serially layer by layer.

Furthermore, the optimality of the results is investigated. The tracking error $e_T$ is given by

$$e_T = \frac{1}{C}\sum_{i=1}^{C}\left\| f(\boldsymbol{\theta}_i) - \boldsymbol{T} \right\|_2, \tag{24}$$

where $\boldsymbol{\theta}_i$ is the result of the $i-$th compute. The manipulability error $e_M$ is given by

$$e_M = \frac{1}{C}\sum_{i=1}^{C}\left(\mu(\boldsymbol{\theta}_{opt}) - \mu(\boldsymbol{\theta}_i)\right), \tag{25}$$

where $\boldsymbol{\theta}_{opt}$ is the approximate optimal solution of manipulability calculated using $L = 5\times10^5$, $P = 1024$. The variance between the solution and the approximate optimal solution can be used to measure the optimality of the solution, which is given by

$$e_{\boldsymbol{\theta}} = \frac{1}{C}\sum_{j=1}^{m}\sum_{i=1}^{C}\left(\theta_{opt}{}^{j} - \theta_i^{j}\right)^2, \tag{26}$$

where $\theta^j$ represents the joint angle of joint $j$. The results are shown in Fig. 3. For non-convex control, the selected parameters have the following goals:

1) Control frequency must be higher than 30Hz;

2) Minimize $e_T$, $e_M$, $e_{\boldsymbol{\theta}}$ when (1) is satisfied;

3) Given $e_{\boldsymbol{\theta}}$ higher priority for the continuity of the results.

In this context, the parameters were chosen as $L = 1000$ and $P = 128$ with the sampling time $\Delta t = 0.028$ s.

### *B. Manipulability Optimization in Dynamic Environments*

In this part, the manipulator performs self-motion manipulability optimization with dynamic obstacles to verify the effectiveness of the framework and the rationality of the selected parameters. The end effector is set as the previous part and the initial configuration is set as $\boldsymbol{\theta} = [0.481, 0.703, 0.093, -0.800, -0.364, 0.992, 0.026]^T$, which is in low manipulability. The sphere obstacle with a radius of 0.15m starts from $[0.00, -0.40, 0.60]^T$ m with a fixed speed of 0.16m/s along the positive Y axis.

Fig. 4 shows the results within 4.5s. The closest distance between the obstacle and the manipulator is presented in Fig. 4(a), which is limited outside the threshold most of the time, and the manipulator successfully performs collision avoidance. In some cases of critical point entry threshold, we speculate it is caused by the discontinuity of the critical point, when it oscillates between links, the restricted critical point is not near the critical point at the next sampling time, and the nearby the critical point at the next sampling time, and the critical point enters the threshold. The manipulability of the self-motion process is presented in Fig. 4(b), the blue dotted line indicates the approximate optimal manipulability at the desired end-effector position. It can be seen that manipulability converges within the first 0.5s and is sacrificed naturally when affected by obstacles. After experiencing dynamic obstacles, the manipulability converges back to the approximate optimal solution, which proves the robustness of the proposed framework. The tracking accuracy is given in Fig. 4(c), which can be seen that the accuracy is on the order of $10^{-6}$m most of the time, and obstacle avoidance has a certain impact on the tracking accuracy. is on the order of $10^{-6}$m most of the time, and obstacle avoidance has a certain impact on the tracking accuracy. The intuitive process of the self-motion can be observed in the video which is available at https://youtu.be/mOKXsy8UqQg.

To further demonstrate the effectiveness of the framework for different geometries, additionally conducted trajectory tracking is shown in Fig. 5, and the video is available at https://youtu.be/OdBe8vXcCBU. The velocity direction of the

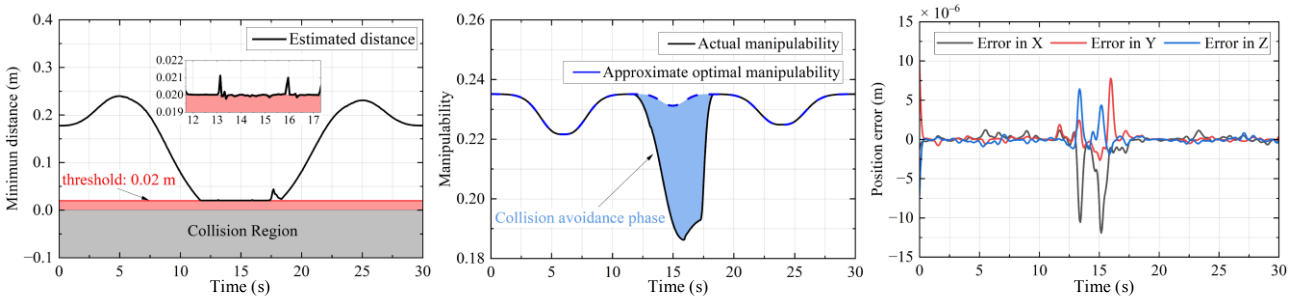


Fig. 6 Results of the fruit sorting experiment. (a) The closest distance between the manipulator and the obstacle. (b) Manipulability. (c) Tracking errors.

TABLE 1 COMPARISONS AMONG OBSTACLE AVOIDANCE AND MANIPULABILITY OPTIMIZATION FRAMEWORKS

| Framework | Planning level | Obstacle avoidance | Dynamic obstacle avoidance | Geometry-based collision detection | Manipulability optimization | Real-time performance |
|---|---|---|---|---|---|---|
| ESM-AMNN | Angle | ✓ | ✓ | ✓ | ✓ | ✓ |
| [22] | Angle | ✓ | ✘ | ✘ | ✓ | ✓ |
| [21] | Acceleration | ✓ | ✘ | ✓ | ✘ | ✓ |
| [9] | Velocity | ✓ | ✘ | ✓ | ✘ | ✓ |
| [23] | Velocity | ✓ | ✓ | ✘ | ✘ | ✓ |
| [34] | Jerk | ✓ | ✓ | ✓ | ✘ | ✘ |
| [17] | Velocity | ✘ | ✘ | ✘ | ✓ | ✓ |
| [15] | Acceleration | ✘ | ✘ | ✘ | ✓ | ✓ |

cube-shaped obstacle changes every 5s.

## V. Experiment and Comparison

In this section, we conduct experiments to verify the effectiveness of the proposed framework. Subsequently, a wide range of comparisons are carried out to highlight our superiority.

### A. Fruit Sorting in Dynamic Environments

In this part, the manipulator performs a fruit grasping task in dynamic environments to examine the feasibility of the proposed framework in the real world, where polygonal obstacles start from $[0.30, -0.30, 0.83]^T$ m with a fixed speed of 0.02m/s along the positive Y axis. The maximum joint velocity was set to 0.7rad/s for safe emergency braking, all other parameters followed the previous section.

The experimental results in 30s are shown in Fig. 6 . The manipulator performs collision avoidance in dynamic environments as shown in Fig. 6 (a). During the tracking process only a very small amount of the closest distance enters the threshold. Fig. 6 (b) shows the manipulability during the tracking process. The approximate optimal manipulability consists of the max manipulability of the end-effector position at each sampling time. It can be seen that except for the obstacle avoidance moment, the manipulability is close to the approximate optimal, which is consistent with the conclusion in the previous section. After obstacle avoidance, the manipulability converges back to the approximate optimal. The tracking accuracy is given in Fig. 6 (c), which is on the order of $10^{-6}$m. The tracking process can be viewed on the video at https://youtu.be/YRZExV6RmrU. The manipulator successfully performs the task, and it can be seen intuitively that the geometry of the obstacle and the manipulator are well taken into account.

### B. Comparison to the State-of-the-Art

In this section, the proposed ESM-AMNN is compared with the state-of-the-art to highlight the superiority of our framework. The work used for comparison comes from the areas of obstacle avoidance and manipulability optimization of redundant manipulators.

The results are shown in TABLE 1. Frameworks that incorporate obstacle avoidance generally do not have the ability to optimize manipulability. [22] can perform those at the same time, but it couldn't detect critical points by geometry and dynamic obstacles are not taken into account. [34] implemented geometry-based obstacle avoidance in dynamic environments, but due to computational complexity, real-time performance cannot be guaranteed. By comparison, the ESM-AMNN is the only online planning framework that integrates both geometry-based dynamic obstacle avoidance and manipulability optimization.

Next, we compare the manipulability of ours and [17], which is generally recognized as one of the state-of-the-art frameworks. Considering the experimental settings in the previous part and ignoring the obstacle avoidance constraints, the starting joint angles are given by ESM-AMNN. The results are shown in Fig. 7, ESM-AMNN performs better and close to the approximate optimal manipulability, we speculate that this is because [17] performs the convex relaxation but the ESM scheme retains the original optimal solution, which further confirms the solving ability and superiority of our framework.

## VI. Conclusion

In this paper, we innovatively propose the ESM-AMNN framework, improving the safety and manipulability of the redundant manipulator in dynamic environments. The ESM scheme integrates dynamic obstacle avoidance, manipulability optimization, trajectory tracking, and joint limits avoidance operating at the joint-angle level. The AMNN controller

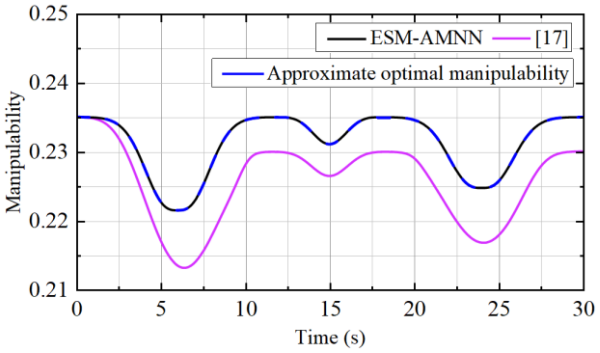


Fig. 7 Comparison of manipulability between frameworks

implements metaheuristic activate functions and GPU parallel computing greatly shortens the computing time of non-convex control. Detailed simulations, experiments, and comparisons demonstrate the framework's effectiveness and superiority. Further directions include incorporating task-based goals and constraints in the scheme, providing the manipulator with more flexible and practical task possibilities.

## REFERENCES


[1] R. E. Jack, "Teaching robots the art of human social synchrony," *Science Robotics,* vol. 9, no. 88, p. eado5755, 2024.

[2] S. Panagou, W. P. Neumann, and F. Fruggiero, "A scoping review of human robot interaction research towards Industry 5.0 human-centric workplaces," *International Journal of Production Research,* vol. 62, no. 3, pp. 974-990, 2024.

[3] D. Liu, C. Li, J. Zhang, and W. Huang, "Robot service failure and recovery: literature review and future directions," *International Journal of Advanced Robotic Systems,* vol. 20, no. 4, 2023.

[4] B. Y. Ma, Z. N. Jiang, Y. Liu, and Z. W. Xie, "Advances in Space Robots for On-Orbit Servicing: A Comprehensive Review," *Advanced Intelligent Systems,* vol. 5, no. 8, p. 2200397, 2023.

[5] R. Zhang *et al.*, "A step towards conditional autonomy-robotic appendectomy," *IEEE Robotics and Automation Letters,* vol. 8, no. 5, pp. 2429-2436, 2023.

[6] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics,* vol. 7, no. 66, p. eabm6074, 2022.

[7] B. Y. Ma, Z. W. Xie, Z. N. Jiang, and H. Liu, "Precise semi-analytical inverse kinematic solution for 7-DOF offset manipulator with arm angle optimization," *Frontiers of Mechanical Engineering,* vol. 16, no. 3, pp. 435-450, Sep. 2021.

[8] H. Shen, W.-F. Xie, J. Tang, and T. Zhou, "Adaptive manipulability-based path planning strategy for industrial robot manipulators," *IEEE/ASME Transactions on Mechatronics,* vol. 28, no. 3, pp. 1742-1753, 2023.

[9] Z. Xu, X. Zhou, H. Wu, X. Li, and S. Li, "Motion planning of manipulators for simultaneous obstacle avoidance and target tracking: An RNN approach with guaranteed performance," *IEEE Transactions on Industrial Electronics,* vol. 69, no. 4, pp. 3887-3897, 2021.

[10] Z. Li and S. Li, "Recursive recurrent neural network: A novel model for manipulator control with different levels of physical constraints," *CAAI Transactions on Intelligence Technology,* Article vol. 8, no. 3, pp. 622-634, Sep. 2023.

[11] T. Yoshikawa, "Manipulability of robotic mechanisms," *the International Journal of Robotics Research,* vol. 4, no. 2, pp. 3-9, 1985.

[12] L. Jin *et al.*, "Perturbed Manipulability Optimization in a Distributed Network of Redundant Robots," *IEEE Transactions on Industrial Electronics,* vol. 68, no. 8, pp. 7209-7220, Aug. 2021.

[13] J. Z. Zhang, L. Jin, and L. Cheng, "RNN for Perturbed Manipulability Optimization of Manipulators Based on a Distributed Scheme: A Game-Theoretic Perspective," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 31, no. 12, pp. 5116-5126, Dec. 2020.

[14] X. H. Lu and Y. M. Jia, "Trajectory Planning of Free-Floating Space Manipulators With Spacecraft Attitude Stabilization and Manipulability Optimization," *IEEE Transactions on Systems Man Cybernetics-Systems,* vol. 51, no. 12, pp. 7346-7362, Dec. 2021.

[15] X. H. Yang, Z. Y. Zhao, B. Y. Ma, Z. C. Xu, J. D. Zhao, and H. Liu, "Kinematic and Dynamic Manipulability Optimizations of Redundant Manipulators Based on RNN Model," *IEEE Transactions on Industrial Informatics,* vol. 20, no. 4, pp. 5763-5773, Apr. 2024.

[16] Y. Ren *et al.*, "Enabling Versatility and Dexterity of the Dual-Arm Manipulators: A General Framework Toward Universal Cooperative Manipulation," *IEEE Transactions on Robotics,* vol. 40, pp. 2024-2045, 2024.

[17] L. Jin, S. Li, H. M. La, and X. Luo, "Manipulability Optimization of Redundant Manipulators Using Dynamic Neural Networks," *IEEE Transactions on Industrial Electronics,* vol. 64, no. 6, pp. 4710-4720, Jun. 2017.

[18] Y. N. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics,* vol. 34, no. 1, pp. 752-759, Feb. 2004.

[19] D. S. Guo and Y. N. Zhang, "Acceleration-Level Inequality-Based MAN Scheme for Obstacle Avoidance of Redundant Robot Manipulators," *IEEE Transactions on Industrial Electronics,* vol. 61, no. 12, pp. 6903-6914, Dec. 2014.

[20] D. C. Chen and Y. N. Zhang, "Minimum jerk norm scheme applied to obstacle avoidance of redundant robot arm with jerk bounded and feedback control," *IET Control Theory and Applications,* vol. 10, no. 15, pp. 1896-1903, Oct. 2016.

[21] B. Y. Ma, Z. W. Xie, B. W. Zhan, Z. A. Jiang, Y. Liu, and H. Liu, "Actual Shape-Based Obstacle Avoidance Synthesized by Velocity-Acceleration Minimization for Redundant Manipulators: An Optimization Perspective," *IEEE Transactions on Systems Man Cybernetics-Systems,* vol. 53, no. 10, pp. 6460-6474, Oct. 2023.

[22] J. W. Tan, M. S. Shang, and L. Jin, "Metaheuristic-Based RNN for Manipulability Optimization of Redundant Manipulators," *IEEE Transactions on Industrial Informatics,* vol. 20, no. 4, pp. 6489-6498, Apr. 2024.

[23] Z. H. Xu, X. F. Zhou, and S. Li, "Deep Recurrent Neural Networks Based Obstacle Avoidance Control for Redundant Manipulators," *Frontiers in Neurorobotics,* vol. 13, Jul. 2019, Art no. 47.

[24] P. B. Yu *et al.*, "An Energy Efficient Soft SIMD Microarchitecture and Its Application on Quantized CNNs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 32, no. 6, pp. 1018-1031, Jun. 2024.

[25] L. Da Luz Dorneles, M. Boiani, and M. Dorn, "Benchmark Problems on GPU: Accelerating Experiments on Metaheuristics," in *2023 IEEE Congress on Evolutionary Computation*, Chicago, 2023.

[26] M. Lupión, N. C. Cruz, J. F. Sanjuan, B. Paechter, and P. M. Ortigosa, "Accelerating neural network architecture search using multi-GPU high-performance computing," *Journal of Supercomputing,* vol. 79, no. 7, pp. 7609-7625, May 2023.

[27] J. Pan, S. Chitta, D. Manocha, and Ieee, "FCL: A General Purpose Library for Collision and Proximity Queries," presented at the 2012 IEEE International Conference on Robotics and Automation (ICRA), 2012.

[28] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *Acm Computing Surveys,* vol. 35, no. 3, pp. 268-308, Sep. 2003.

[29] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Information Sciences,* vol. 237, pp. 82-117, Jul. 2013.

[30] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 4th ed. Boston, USA: Addison-Wesley Longman Publishing Co. Inc., 1988.

[31] L. Y. Wang, Q. J. Cao, Z. X. Zhang, S. Mirjalili, and W. G. Zhao, "Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence,* vol. 114, Sep. 2022, Art no. 105082.

[32] F. J. Solis and R. J. B. Wets, "Minimization by Random Search Techniques," *Mathematics of Operations Research,* vol. 6, no. 1, pp. 19-30, 1981.

[33] S. Haddadin *et al.*, "The Franka Emika Robot A Reference Platform for Robotics Research and Education," *IEEE Robotics & Automation Magazine,* vol. 29, no. 2, pp. 46-64, Jun. 2022.

[34] Y. L. Wen and P. Pagilla, "Path-Constrained and Collision-Free Optimal Trajectory Planning for Robot Manipulators," *IEEE Transactions on Automation Science and Engineering,* vol. 20, no. 2, pp. 763-774, Apr. 2023.