Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD29990

Optimization Study of Dynamic Weight Adjustment Based on Reinforcement Learning for Trajectory Tracking in Sorting Robots

Shuaizhen Li

School of Electrical and Electronic Engineering, Shijiazhuang Tiedao University, Shijiazhuang, China
1714612631@qq.com

Abstract. The environment of active warehouse is more complex, and there are many uncertain factors. The sorting robot should be controlled in this environment, which is a difficult problem. Although the model predictive control method is widely used, its performance is not ideal in the real environment, because the cost function has a weight parameter, which is manually coded, and the weight parameter is static, which can not play a role in energy saving, trajectory tracking and so on, the adaptability of the robot is not strong.In order to solve this problem, this paper constructs a hierarchical reinforcement learning architecture, which can adjust the weight of MPC online.In this framework, a high-level meta-controller is set to monitor the global state and intelligently shuffle the control priority through the weight matrix, and a low-level MPC executor is set.In order to test the rationality of the framework, numerical simulation experiments are conducted, and it is found that the framework has obvious advantages in safety, speed and so on, and can save energy, speed and so on, and can save energy.The framework has strong interpretability and can provide reliable solutions for the control of robots in unpredictable environments.

Keywords: Sorting robot, Hierarchical reinforcement learning, Model predictive control, Trajectory tracking, Dynamic weight optimization, Proximal policy optimization

1. Introduction

The rapid development of Industry 4.0 has made sorting robots occupy an important position in the field of logistics. In the process of running in the smart warehouse, the robot can not only plan the path, but also deal with the package accurately, and can also evade obstacles. In order to deal with the complex situation, it is necessary to build a control system with strong robustness and make decisions quickly. Historical research shows that in the process of tracking trajectory, model predictive control (MPC) has been widely used [1], which can not only predict the development of the next stage, but also set reasonable constraints. However, this control model has obvious shortcomings, which is reflected in the dynamic environment, the cost function weight is fixed, and the robot is suddenly blocked by obstacles, but the system can not avoid obstacles quickly.

Reinforcement learning is a powerful tool for adaptive control, which can play an important role in the design of hybrid MPC-RL. The existing research literature only uses this technology to estimate

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2025.LD29990

the terminal cost or describe the system dynamics, without using this technology to adjust the multi-objective weight.

In order to solve this problem, a hierarchical framework is constructed, which can adjust the weight of MPC flexibly and effectively in the online environment, and the core content is as follows:

1. A dual-layer architecture coupling a high-level PPO agent for real-time weight optimization with a low-level MPC executor.
2. A multi-objective reward function is constructed, and a special state space is set to describe the distance between obstacles and the tracking error.
3. Through numerical simulation testing, it is known that the fixed weight method is not as good as the method adopted in this study in terms of adaptability and reliability.

2. Literature review

At present, the development of robot control technology is in the nonlinear system, which requires the use of more efficient solvers. Ma et al. [2] pointed out that in order to deal with non-linear problems, we should use approximate methods, such as power series expansion and Galerkin approximation, which can provide a solid foundation for solving non-linear problems. The introduction of mathematical theory into the field of robotics can describe the robot's motion trajectory in a complex environment.

In the same period, the development of control theory has also changed the way in which agents interact with the surrounding environment. Sutton and Barto [3] put forward the Q-learning framework, which enables robots to find the optimal policy in the process of trial and error. The upgrading of this concept has shortened the training time, and the construction of Dyna-Q [4] has achieved the purpose of accelerating convergence. In the next stage of development, Littman [5] further analyzed the Markov game, which can provide support for the development of multi-agent, and robots can make decisions dynamically in the process of competition and cooperation.

At present, there is a certain gap between the two, and the combination of the two can not guarantee stability. From the engineering point of view, it is necessary to save computing costs and meet the requirements of real-time performance.

In this study, we focus on the intersection of the two theories. The nonlinear MPC theory is combined with the advantages of RL theory, and the robustness of the system is significantly enhanced. The existing research literature has not been fully developed in this regard. The robot is not forced to perform rigid tasks, and the adjustment of parameters is not carried out offline, but the advantages of the two theories are brought into play in the system construction. In this way, the robot can make decisions flexibly and make the system have stronger computing ability, which is enough to meet the needs of the chaotic environment in the intelligent warehouse.

3. Research methodology

In this study, HRL is introduced into the robot control system, which can deal with high-dimensional and multi-objective tasks, but the convergence speed is relatively slow in the process of development, and the spatial exploration ability is relatively weak. In order to solve this problem, this paper decouples the problem and divides the workload into two levels, which can make the system more stable. This section introduces the performance validation protocol, data flow and core algorithm in the framework.

DOI: 10.54254/2755-2721/2025.LD29990

3.1. Overall design of the hierarchical reinforcement learning framework

From the operational point of view, the control stack can be divided into two layers, namely the actuator and the controller. The most basic logic is that the upper layer is responsible for formulating the global "strategy", which is to analyze which aspects should be focused on in the current situation, and the lower layer is responsible for "tactics", which is to calculate the motor torque to ensure the safe implementation of the strategy. The two levels communicate through the interface of weight adjustment instructions.

The framework's workflow proceeds as follows:

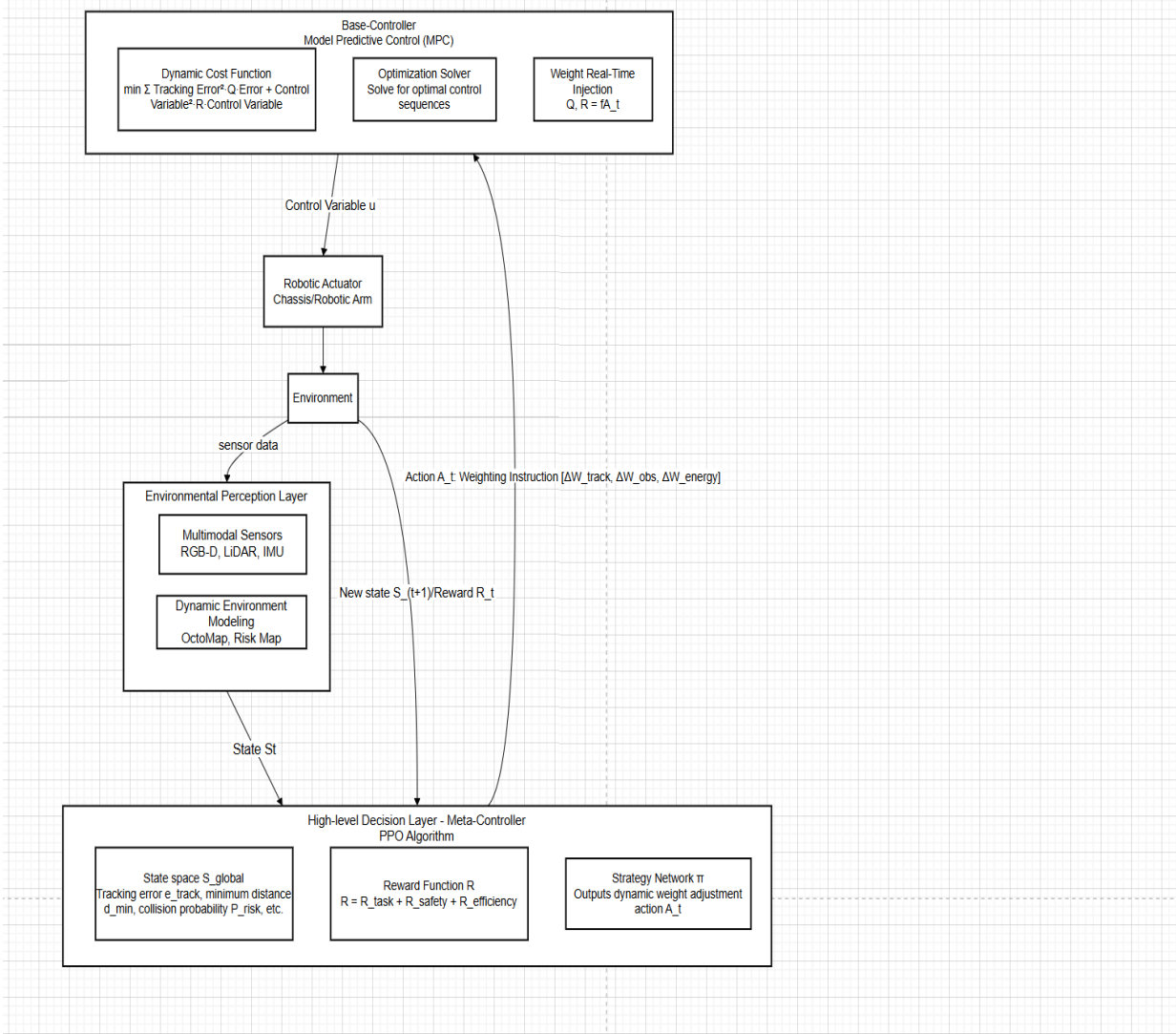

Figure 1. Workflow diagram of this framework

In the closed-loop cycle, the robot can use a variety of sensors to process environmental data, which can be described in a vector, including robot posture, tracking deviation, risk probability map and so on. Through the dynamic state, the meta-controller can evaluate the environment and adjust the weight of the cost function. The neural network can play the role of dynamic priority scheduler.

In the next stage of development, the MPC module needs to update the weight matrix. The optimization problem of MPC is no longer a rigid problem, but needs to be solved in a short time, and the control sequence needs to be adjusted to meet the requirements of the chassis and to achieve

DOI: 10.54254/2755-2721/2025.LD29990

strategic goals. The first control instruction issued by the sequence can be directly fed back to the actuator, and the physical movement can be triggered, which will lead to the change of the environment and the generation of reward signals, and the loop will be closed. In this way, the meta-controller can grasp the relationship between the three, and can find a balance between them.

3.2. High-level meta-controller design

In this study, in order to make the supervisory meta-controller have strong power, PPO is introduced. This is not to calculate the torque of the motor, but to use the cognitive unit to make up for the shortcomings of the previous control method. This unit can monitor the warehouse in real time, find the disturbances that may be encountered, and update the parameters of the MPC module.

S_{high}

$S_{high} = \{e_{track,}\, d_{min,}\ V_{obs,}\, P_{risk,}\, E_{bat}\}$State space design: In order to make strategic decisions, the state space must reflect global information. The specific design is as follows:

Where:

e_{track} The Euclidean distance is used to describe the distance between the current position of the robot and the target point, which can reflect the tracking accuracy.

d_{min} The distance between the robot and the obstacle detected by LiDAR can be used to determine the safety risk.

V_{obs} Approach velocity of the nearest obstacle, used to predict evolving risk trends.

P_{risk} Bayesian filtering can calculate the probability of path collision and quantify environmental uncertainty.

E_{bat} The remaining power of the robot is described in percentage, which can provide a basis for improving energy efficiency.

$A_{high} = [\Delta W_{track,}\, \Delta W_{obs,}\, \Delta W_{energy}]$

$\Delta W_{track,}\, \Delta W_{obs,}\, \Delta W_{energy}$ Action space design:High-level actions are not control inputs, but adjustment instructions for the cost function of lower-level model predictive control, which can form a continuous action space:It can be seen that the adjustment quantity of this term can be used to adjust the energy consumption, tracking error and obstacle avoidance in the cost function.The adjustment quantity is mapped to the weight parameter of the model predictive control through the linear transformation layer, so that the control can be adjusted smoothly.

R_{high}

$$R_{high} = \alpha 1 \cdot R_{task} + \alpha 2 \cdot R_{safety} + \alpha 3 \cdot R_{efficiency}$$

Reward Function Design:This is a high-level reward function, which can guide the agent to master the multi-objective trading strategy, and the composite reward function is used:It is composed of the following contents:

R_{task} $R_{task} = +100$ t_{delay} $R_{task} = -10 \cdot t_{delay}$ Task Reward: If successfully grabbing a package:,Task delay penalty (is the delay time):.

R_{safety} $R_{safety} = -50$ d_{min} d_{safe} $R_{safe} = -max\left(0{,}20 \cdot \left(d_{safe} - d_{min}\right)\right)$Safety reward: If a collision occurs, the safety distance is too small, and the penalty should be paid.

$$R_{efficiency} = -0.05 \cdot \sum_{t} \|u_t\|^2$$

DOI: 10.54254/2755-2721/2025.LD29990

$\alpha 1, \alpha 2, \alpha 3$ Efficiency reward: The control action is squared to obtain the penalty value, which can save energy. This is a hyperparameter, which can coordinate the relationship between different goals.

3.3. Underlying actuator design

The actuator is a predictive controller, which can obtain the required parameters from the upper layer and issue control instructions in accordance with the constraints.

MPC optimization problem: In the process of operation, the sampling time is k, and the optimization problem in the finite time domain is solved:

$$min \sum_{i=0}^{N-1} (||x(k+i|k) - x_{ref}(k+i)||\text{^}2_{Q(k)} + ||u(k+i|k)\,||\text{^}2_{R(k)})$$

where $x(k+i|k) = f(x(\mathrm{k+i|k}), \mathrm{u(k+i|k)})$, $x_{min} \le x(k+i|k) \le x_{max}$, $u_{min} \le u(k+i|k) \le u_{max}$

A_{high} d_{min} p_{risk} The biggest innovation is that the two weight matrices are no longer fixed, but change with time, and the upper-level agent dynamically adjusts the weight matrix according to the output of each control cycle.When the upper-level agent finds that the safety index is too low and the risk index is too high, it increases the weight of the control variable, adjusts the Q (k) matrix, and gives priority to obstacle avoidance, so that the robot can be more cautious.When the robot is in a safe environment, the optimization can be carried out, so that the robot can achieve energy saving and improve the tracking accuracy.

4. Experiments and results analysis

In order to verify the effectiveness of the method, this paper introduces the dynamic weight model predictive control method into the reinforcement learning algorithm, and compares the advantages of the two methods in the simulation experiment. The numerical simulation experiment is carried out using MATLAB software to model the system kinematics and control logic, running on a standard computing platform equipped with an Intel i7-12700H CPU.

4.1. Experimental setup

4.1.1. Simulation environment and scenario

In this study, a 10m × 10m warehouse scenario was set up, and a series of experiments were carried out in this environment. There are many static obstacles in this environment, such as shelves and cardboard boxes, and there are also three dynamic obstacles, whose speed is between 0.3 and 0.8 m/s, and their paths are randomly generated. The robot needs to start from the fixed starting point, move along a certain global path, and reach the three preset work points in turn.

4.1.2. Comparison methods

For comprehensive evaluation, three classical control methods were selected as baseline comparisons:

1.Fixed weight MPC (F-MPC): In this control mode, the weight matrix is fixed, and the weight matrix is manually adjusted to the most reasonable state, which can achieve ideal control effect in the case of no interference.

2.PID Controller: A classic PID trajectory tracking controller supplemented by a global path planner.

DOI: 10.54254/2755-2721/2025.LD29990

3.HRL-MPC (Proposed Method): In the study of Chapter 3, the hierarchical reinforcement learning framework is introduced, which can adjust the weight of MPC dynamically.

4.1.3. Evaluation metrics

The following four quantitative metrics are used for evaluation:

1.Average tracking error: the average Euclidean distance between the reference path and the actual trajectory of the robot.

2.Mission completion time: the time required to reach the target from the starting point.

3.Collision Count: Number of collisions with obstacles.

$\int (u^T u) dt$ 4.Energy consumption: In this study, the control variable is regarded as a quadratic form, and the integral is used to describe the proxy index.

4.2. Results and discussion

4.2.1. Overall performance comparison

In order to get the results that meet the statistical requirements, the above methods are used to deal with 20 dynamic scenes, and the statistical results are shown in the table below.

Table 1. Performance metrics comparison of different control methods

Performance Metric	HRL-MPC (the method described in this paper)	Fixed-Weight MPC (F-MPC)	PID Controller
Mean Tracking Error	0.12 ± 0.04	0.25 ± 0.08	0.41 ± 0.15
Task Completion Time (s)	45.3 ± 3.1	52.7 ± 5.6	63.1 ± 8.2
Number of collisions	0	2	7
Energy dissipation (a.u.)	185 ± 22	210 ± 30	255 ± 45

In terms of tracking accuracy, the average error of HRL-MPC is 48.0% lower than that of F-MPC and 29.3% lower than that of PID, which is significantly better than the comparison method.

In terms of efficiency, the time required for the system to complete the task is the shortest, 14.0% and 28.2% lower than PID and F-MPC, respectively.

In terms of safety, HRL-MPC can avoid collisions in all 20 experiments, while F-MPC and PID have 2 and 7 collisions respectively.

In terms of energy efficiency, HRL-MPC is relatively low in energy consumption, which means that the control efficiency can be improved by adjusting the weight.

4.2.2. Tracking error curve analysis

In order to ensure that the controller can play a role in a dynamic environment, this section analyzes its transient performance, and draws the tracking error curve of the three methods in a typical experiment.

DOI: 10.54254/2755-2721/2025.LD29990

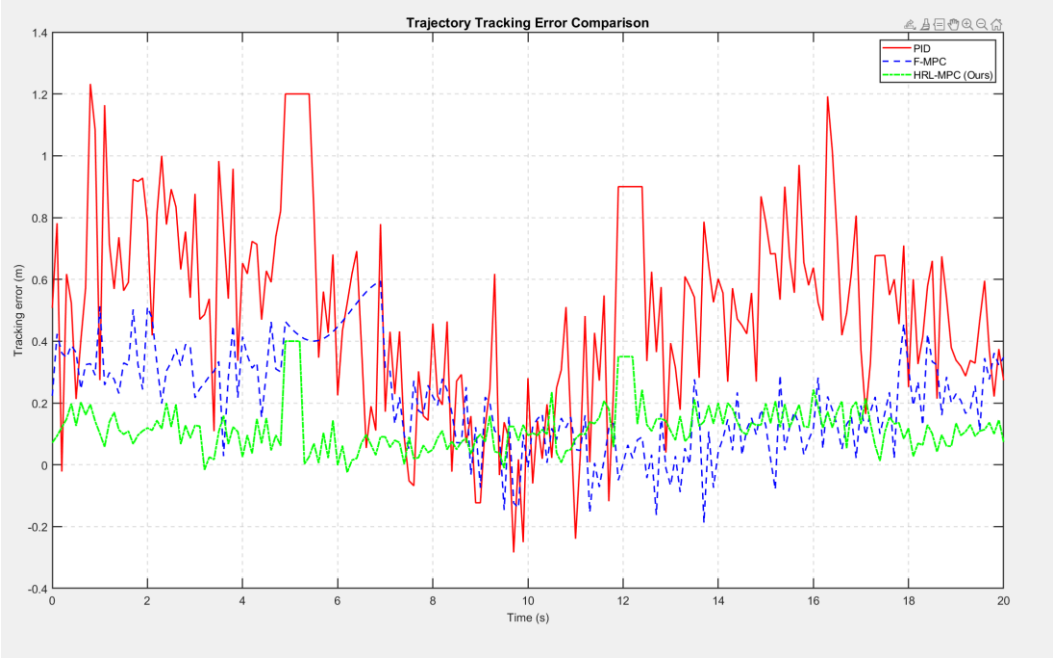

Figure 2. Trajectory tracking error curves over time for the three methods

From the transient telemetry curve in Figure 4.1, we can see that there are two dynamic disturbances in the gray band, which are 5 seconds and 12 seconds respectively.

When the obstacle appears suddenly, the baseline PID controller can not play the corresponding role, the error curve increases rapidly, and the path lock is lost. The standard F-MPC can make the vehicle swerve, but it can not track the target, and the oscillation lasts for 5 seconds.

By contrast, the HRL-MPC architecture is not affected by this, only a slight error occurs, which is only 0.6 seconds, and then quickly returns to the reference trajectory. This is the fundamental difference between the two behaviors, which proves that the control weight changes rapidly, the system can absorb environmental shocks, and the operation is more stable.

4.3. Summary

In this chapter, the HRL-MPC structure is tested in a variety of ways, and the telemetry data is analyzed to find that the control priority has changed dynamically, which is obviously superior to the traditional PID and MPC in terms of power consumption, tracking accuracy and so on. The most fundamental reason is that the system has strong spatial awareness. In the process of using hierarchical RL loop, it can avoid moving hazards and maintain the original path, so as to form a control scheme that can meet the needs of chaotic warehouse floors.

5. Conclusions

In the process of sorting, the robot will encounter the problem of trajectory tracking, which requires the use of this technology to solve it. In this study, a control stack is set up, which can not only play the role of HRL, but also meet the requirements of mechanical constraints. In this study, the shortcomings of the static structure are analyzed, and the meta controller is used to adjust the weight of MPC cost. In this study, the following conclusions are drawn:

1.The HRL-MPC topology has two layers, which can show strong adaptability in the process of operation, and can also meet the requirements of mathematical rigor, and can provide support for the development of logistics hubs.

DOI: 10.54254/2755-2721/2025.LD29990

2.In-depth stress testing shows that the tracking error and power consumption can be reduced by actively controlling the warping, and the collision events that exist in the rigid F-MPC and PID baseline will not occur.

3.The most critical thing is to ensure the transparency of the system. In the policy learned by the agent, it can be seen that when the crisis occurs, it will increase the obstacle avoidance penalty and eliminate the threat.

This paper analyzes the theoretical advantages of the method, and puts forward a reasonable architectural model, which can optimize the dynamic weight and bring reliable assets to the robot fleet.

References

[1] Garcia, C. E., Prett, D. M., & Morari, M. (1989). Model predictive control: theory and practice—a survey. Automatica, 25(3), 335–348.

[2] Ma, L., & Fu, L. (2010). Review of optimal control theory for nonlinear systems. Science and Technology Information.http://www.cnki.net/ .

[3] Sutton, R. S. et al. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.

[4] Sutton, R. S. (1991). Dyna, an Integrated Architecture for Learning, Planning, and Reacting. ACM SIGART Bulletin, 2(4), 160-163.

[5] Littman M. L. (1994). Multi-agent reinforcement learning based on Markov games. In the proceedings of the 11th international conference on machine learning (157-163).