Adv. Manuf.
https://doi.org/10.1007/s40436-024-00529-6

BRMPNet: bidirectional recurrent motion planning networks for generic robotic platforms in smart manufacturing

Bo-Han Feng[1] · Bo-Yan Li[1] · Xin-Ting Jiang[1] · Qi Zhou[1] · You-Yi Bi[1]

Received: 29 September 2023 / Revised: 14 December 2023 / Accepted: 2 September 2024

Abstract In the era of Industry 4.0, robot motion planning faces unprecedented challenges in adapting those high-dimension dynamic working environments with rigorous real-time planning requirements. Traditional sampling-based planning algorithms can find solutions in high-dimensional spaces but often struggle with achieving the balance among computational efficiency, real-time adaptability, and solution optimality. To overcome these challenges and unlock the full potential of robotic automation in smart manufacturing, we propose bidirectional recurrent motion planning network (BRMPNet). As an imitation learning-based approach for robot motion planning, it leverages deep neural networks to learn the heuristics for approximate-optimal path planning. BRMPNet employs the refined PointNet++ network to incorporate raw point-cloud information from depth sensors and generates paths with a bidirectional strategy using long short-term memory (LSTM) network. It can also be integrated with traditional sampling-based planning algorithms, offering theoretical assurance of the probabilistic completeness for solutions. To validate the effectiveness of BRMPNet, we conduct a series of experiments, benchmarking its performance against the state-of-the-art motion planning algorithms. These experiments are specifically designed to simulate common operations encountered within generic robotic platforms in smart manufacturing such as mobile robots and multi-joint robotic arms. The results demonstrate BRMPNet's superior performance on key metrics including solution quality and computational efficiency, suggesting the promising potential of learning-based planning in addressing complex motion planning challenges.

Keywords Robot motion planning · Imitation learning · Deep neural network · Smart manufacturing · Adaptive and real-time planning

1 Introduction

As the fourth industrial revolution advances, robots are transitioning to the key players in the landscape of smart manufacturing [1]. Generic robotic platforms such as automated guided vehicles (AGVs), multi-joint robotic arms, and their combinations (i.e., hybrid robots) are playing increasingly crucial roles in smart factory as shown in Fig. 1. When equipped with specific end-effectors such as gripper, screwdriver or camera, these robotic platforms can take various tasks from material handling, precise assembly to production inspection [2, 3].

A diverse range of tasks mean more complex working environments for robots. Besides avoiding regular static obstacles like raw materials and equipment, robots must also adapt to dynamic environments where interactions with human operators, running machinery or other moving robots occur frequently. For example, consider a robotic arm in a flexible assembly line: it must efficiently navigate around obstacles such as fixtures or material boxes while also adapting its motion in real-time to cope with shifts in component and part sizes, alterations in workbench configurations, and changes in production schedules. An absence of real-time motion planning for robots not only jeopardizes the safety of equipment and human workers, but also risks creating a chain reaction such as disturbing production schedule with increasing manufacturing cost. Therefore, the development

✉ You-Yi Bi
youyi.bi@sjtu.edu.cn

[1] University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, Shanghai 200240, People's Republic of China

Published online: 11 November 2024

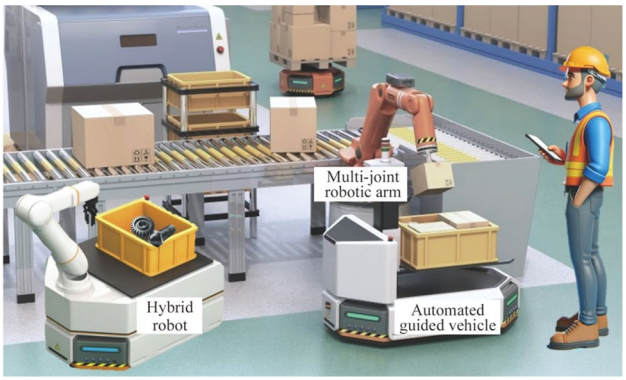

Fig. 1 Generic robotic platforms in smart factory

of motion planning methods with real-time dynamic adaptability is significant for robots to strive for both safety and efficiency in smart manufacturing landscapes.

Motion planning aims to compute path that allows the robot to move from a starting position to a target destination while avoiding collisions with obstacles. Traditional motion planning algorithms can be classified into three categories: search-based, sampling-based, and optimization-based. Search-based algorithms rely on constructing a graph from the robot's configuration space and then employing classic search techniques (e.g., A*) to find a path [4]. These methods guarantee exact solutions but are computationally expensive when dealing with high-dimensional configuration spaces. Their dependency on discrete space often leads to a loss in solution fineness, potentially overlooking narrow but valid paths. Sampling-based algorithms, such as rapidly-exploring random trees (RRTs) [5] or probabilistic roadmaps (PRMs) [6], focus on randomly sampling the configuration space and connecting these sample points to form paths. They are probabilistically complete to find solutions with the probability of 1 if the path exists. But under time limitations in manufacturing, their random sampling nature may produce irregular paths or simply fail to find solutions. Optimization-based algorithms take an initial path, potentially infeasible, and iteratively refine it to avoid obstacles while optimizing certain criteria such as path length [7]. These algorithms require careful tuning of the cost functions to ensure convergence and are prone to obtain local optima. Furthermore, non-linear optimization is computationally expensive and slow for difficult planning problems.

As robots' operational environments in smart manufacturing grow more dynamic and complex, traditional motion planning algorithms become increasingly inapplicable. Their major limitation is the inability to adapt and generalize across diverse working environments. The scalability and computational efficiency of traditional algorithms are also called into question when dealing with real-time motion planning needs in intricate environments. Recently, researchers are turning to learning-based motion planning algorithms [8, 9]. Learning-based algorithms, with their inherent capability to learn and generalize from vast experiences, offer the exciting prospect of motion planners that can adapt to new environments without explicit reprogramming for each new environment [10]. They can handle high-dimensional spaces to the level of efficacy that traditional algorithms may struggle to achieve. This adaptability is especially beneficial for generic robotic platforms in smart manufacturing, which often have to switch between different tasks and environments with limited downtime for reconfiguration. However, existing learning-based algorithms still struggle with achieving well generalizability, real-time adaptability, and success rate assurance in unknown environments.

Thus, in this paper, we introduce the bidirectional recursive motion planning network (BRMPNet) to meet the growing complexities and demands of robotic applications in manufacturing. The core idea of our approach is to harness the capabilities of neural networks to perform environment feature encoding and construct a motion planning architecture with high adaptability and real-time decision-making ability. Compared to existing learning-based algorithms, our approach is more scalable, computationally efficient, and robust to environmental changes. The main contributions of our research include.

(i) An end-to-end neural network architecture for motion planning is proposed to operate on point cloud of the environment and generate near-optimal paths while avoiding obstacles. This architecture is highly adaptive, enabling real-time response to dynamic changes of the environment.
(ii) A refined PointNet++ network is integrated to maximize the utility of point-cloud information collected from depth sensors.
(iii) A long short-term memory (LSTM) network is employed to capture the dynamic temporal dependencies inherent in the planning process. This treatment can iteratively update the motion plan based on current environmental information and historical planning data, achieving real-time adaptability.
(iv) A hybrid online execution mechanism is designed to maintain the compatibility of our learning-based approach with traditional sampling-based planning algorithms. This mechanism provides probability guarantees on solution completeness, ensuring a reliable and robust motion planning performance in various applications.

The effectiveness of the proposed approach is examined in both computer simulations and physical robot experiments. Results show that our approach has significant

advantages over other state-of-the-art methods in terms of computational efficiency and solution quality. The rest of the paper is structured as follows. Section 2 presents a literature review of learning-based motion planning algorithms. Section 3 provides the problem formulation, and Section 4 introduces the proposed approach and explains the key techniques involved. Section 5 showcases the effectiveness of the proposed approach through experiments. Section 6 provides the summary of this work and highlights potential directions for future research.

2 Related work

In recent years, learning-based methods have attracted considerable attention, emerging as a class of novel and versatile motion planning solutions for robotic systems. Characterized by their high adaptability and real-time decision-making ability, these methods harness the power of machine learning, particularly deep neural networks, to model and solve high-dimensional, non-linear planning problems that are computationally intractable for traditional algorithms. Learning-based motion planning methods can be generally classified into two categories: end-to-end and module replacement [11]. Representative methods in these two categories are reviewed and discussed in the following subsections.

2.1 End-to-end methods

End-to-end methods aspire to supplant the entire traditional motion planning pipeline. They leverage deep neural network architectures to generate end-to-end solutions, furnishing collision-free paths in each configuration space without intermediate steps. For instance, Pfeiffer et al. [12] developed a convolutional neural networks (CNN)-based navigation model that enabled robots to map raw 2D-laser environment data and target positions directly to steering commands. Hamandi et al. [13] introduced DeepMotion, a human-aware navigation model that combined CNN and LSTM network layers to process laser data for safe navigation through crowds.

In the quest for iterative, end-to-end collision-free paths, several algorithms have emerged. Bency et al. [14] created OracleNet, which utilized recurrent neural networks (RNNs) for rapid and near-optimal motion planning in static, high-dimensional settings. Ichter and Pavone [8] proposed latent sampling-based motion planning (L-SBMP), a technique that integrated autoencoding, dynamics, and collision-checking networks to learn a plannable latent representation of robotic systems. Qureshi et al. [9] introduced motion-planning networks, which encoded environmental data into a latent space and generated predicted collision-free paths between start and goal configurations. Fishman et al. [15] proposed motion policy networks, an end-to-end model trained on over three million planning problems to produce collision-free and smooth motions. Inspired by generative neural networks, Kurutach et al. [16] presented a causal InfoGAN model that learnt to produce feasible observations to guide motion. Huh et al. [17] introduced a neural network-based cost-to-go function for manipulator motion planning using workspace sensor inputs.

Deep reinforcement learning (DRL) also offers another innovative avenue. It enables robots to learn planning paths through trial and error, guided by the principles of reward optimization. In DRL, two main components are often highlighted: the policy function and the value function. The policy function dictates the robot's actions at any given state, while the value function estimates the expected future rewards for those actions. By optimizing these components, DRL algorithms [18–20] have also been applied in motion planning. However, DRL algorithms face challenges including high computational cost due to data-intensive training and sensitivity to reward function design that can lead to failed planning.

2.2 Module replacement methods

Module replacement methods focus on refining specific modules within traditional motion planning algorithms. Typical modules include the preprocessing, prediction, execution, and collision-checking. Various deep learning architectures, such as CNN, LSTM, DRL, graph neural networks (GNNs), and generative networks, etc., have been successfully employed for this purpose.

The preprocessing module aims to generate processed configuration space in advance to guide the expansion of searching in traditional motion-planning algorithms. To refine this module, Khan et al. [21] utilized GNN to encode the topology of configuration space and calculate sampling distribution parameters. Similarly, Wang et al. [22] introduced Neural RRT*, which combined a pretrained CNN with RRT* to guide the sampling process and improve path planning performance. In the context of dual-arm assembly robots, Ying et al. [23] employed LSTM network to provide informed sampling points for bidirectional-RRT. Additionally, Kumar et al. [10] introduced LEGO, an algorithm that used conditional variational auto-encoders (CVAEs) to improve roadmap generation in sampling-based motion planning.

The prediction module is to anticipate the structure of the unknown space and feasible path in the motion planning process. To improve this module, Qureshi and Yip [24] combined the Contractive AutoEncoder with a stochastic neural network to enable informed sample generation. Elhafsi et al.

[25] employed a conditional neural process (CNP) architecture to forecast unobserved environmental terrains.

The execution module generates a new state by taking actions. Several key contributions have emerged to improve this module. For instance, Kim and An [26] developed a learning heuristic A* algorithm that employed neural networks to improve the efficiency of graph-based search. Similarly, Guzzi et al. [27] designed a data-driven planning algorithm that integrated a state estimator with a sampling-based planner to optimize path generation.

The collision-checking module examines the safety of the generated path. Several contributions have been made to refine this module. For example, Chase et al. [28] introduced ClearanceNet, a collision checking heuristic network and a planning algorithm CN-RRT which leveraged ClearanceNet's capabilities for efficient motion planning. Zhang et al. [29] used DRL to develop a rejection sampling model and optimized the sampling distribution to reduce the number of collision detection by a policy gradient approach. Tran et al. [30] formulated a framework that combined Convolutional Autoencoders with Multi-layer Perceptrons to predict sample collisions effectively. Yu and Gao [31] utilized GNN for path exploration and path smoothing to significantly alleviate the computational burden of collision checking in sampling-based motion planning.

2.3 Summary of existing learning-based motion planning methods

While the aforementioned methods have made significant strides in learning-based motion planning, there are still challenges to be addressed, particularly in the context of smart manufacturing where robots often need to plan their motions rapidly to adapt to high-dimensional dynamic spaces and flexible jobs. The increasing need for efficient fusion of multi-sensor data and strict safety constraints for human-machine interaction brings more complexity for motion planning that existing methods cannot fully handle. Also, existing end-to-end methods often lack the flexibility to be integrated with traditional motion planning algorithms, limiting their applicability in diverse environments. Module replacement methods, on the other hand, tend to refine specific stages of the planning pipeline without offering a comprehensive solution. Moreover, most existing methods do not fully exploit the rich data available from advanced depth sensors, which is crucial for real-world applications in smart manufacturing.

To fill these gaps, we introduce BRMPNet, a comprehensive and adaptable solution for robot motion planning. It extends the principle underlying end-to-end methods, and can easily switch to module replacement methods under necessary conditions, thus bridging the advantages of two principal categories of learning-based motion planning methods.

3 Problem formulation

The configuration space $\mathcal{C}$ of robot represents the set of all potential states, characterized by a given degree of freedom for the robot. Let $\mathcal{C}$ be a subset of $\mathbb{R}^d$,where $d \in \mathbb{N}, d \geq 2$. And the robot's workspace $\mathcal{W}$, the physical area where it interacts with the environment, serves as the real-world instantiation of these abstract configurations, and is represented as a subset of $\mathbb{R}^m$, where $m \in \{2, 3\}$ indicating the workspace's dimensional characteristics. Let $\mathcal{C}_{\text{obs}}$ be the obstacle region of configuration space. The representation of obstacle spaces $\mathcal{C}_{\text{obs}}$ is typically not pre-defined. Instead, a collision-checker is used to determine if $\mathcal{C}$ intersects with any obstacle $\mathcal{W}_{\text{obs}}$ in the workspace $\mathcal{W}$. The obstacle-free configuration space is denoted as $\mathcal{C}_{\text{free}} = \text{cl}\big(\mathcal{C} \setminus \mathcal{C}_{\text{obs}}\big)$, where $\text{cl}(\cdot)$ denotes the closure of a set. The start point x_{start} and goal point x_{goal} are both elements of $\mathcal{C}_{\text{free}}$. A planning problem is defined by a triplet $(x_{\text{start}}, x_{\text{goal}}, \mathcal{C}_{\text{free}})$. The definitions of key concepts in robot motion planning, such as feasible path planning and probabilistic completeness are provided as follows.

Definition 1 (Feasible path planning)
Given a path planning problem $(x_{\text{start}}, x_{\text{goal}}, \mathcal{C}_{\text{free}})$, find a feasible path $\sigma : [0, T] \to \mathcal{C}_{\text{free}}$ such that $\sigma(0) = x_{\text{start}}$ and $\sigma(T) = x_{\text{goal}}$, if one exists. If no such path exists, report failure.

Definition 2 (Probabilistic completeness)
Given any feasible path planning problem $(x_{\text{start}}, x_{\text{goal}}, \mathcal{C}_{\text{free}})$, the motion planning algorithm is probabilistic completeness if the probability which returns a feasible path goes to one as the number of planning attempts goes to infinity,

$$\liminf_{n\to\infty} P\big(\sigma_n \neq \emptyset\big) = 1, \tag{1}$$

where n is the number of planning attempts and $\sigma_n \subset \sigma_{\text{feasible}}$ is the set of feasible paths found by the planner from those attempts.

4 Methods

4.1 Model architecture

Figure 2 shows the overall architecture of BRMPNet, which integrates a refined PointNet++ network with an LSTM network. The workflow initiates with the refined PointNet++ network processing the point cloud data to extract critical environmental features, which are then passed to the LSTM network for path generation. Imitation learning is incorporated within the BRMPNet, where the learning-based approach is trained to mimic expert-provided paths, allowing for the generation of near-optimal paths in unknown

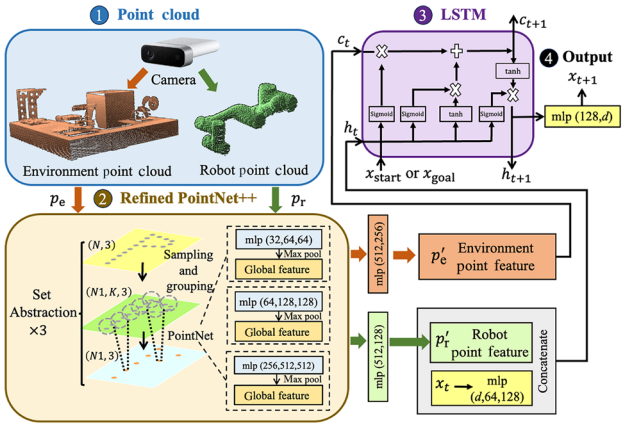

Fig. 2 BRMPNet architecture

environments. The refined PointNet++ network and the LSTM network are particularly designed for sophisticated feature extraction from environment information (i.e., point cloud data), and generating paths, respectively. The details regarding their structures are provided as follows.

4.1.1 Refined PointNet++ network

The reason for refining PointNet++ [32] stems from its superior performance in encoding environmental variables compared to traditional approaches such as MLP, VoxelNet [33], and its predecessor, PointNet [34]. Several salient features contribute to its performance. For example, the hierarchical structure of PointNet++ excels at capturing local spatial patterns within point clouds, thereby enabling learning at multiple scales of point clouds. By processing point clouds directly, PointNet++ can maintain the fidelity of environment data without resorting to voxelization and relying on the specific ordering of points. Also, its ability to manage variations in density and scale of point clouds not only reduces computational overhead but also preserves high-resolution details. These features collectively make PointNet++ a suitable choice for encoding environmental data in robot motion planning.

To further enhance its effectiveness of environmental feature extraction for motion planning, we refine the PointNet++ structure. We first remove the segmentation and classification networks in the original PointNet++ architecture. This decision is driven by the need to streamline the architecture for specific applications in robot motion planning, where the primary focus is on precise spatial feature extraction rather than object identification or categorization. In addition, we improve the design of the set abstraction layers in original PointNet++ by changing the number of neurons, sampling size, grouping scope and other related parameters in these layers to ensure the point cloud information from generic robotic platforms can be processed more efficiently.

Our refined PointNet++ structure is composed by three set abstraction levels. Each level is meticulously engineered to enhance the extraction and processing of spatial features. At each set abstraction level, a set of points is processed and abstracted to produce a new set with fewer elements. The set abstraction level is made of two key operations: sampling and grouping operation, and PointNet operation. The sampling and grouping operation pick a subset of $N \times 3$ points from the input to serve as local region centroids, and subsequently identify neighboring points around these centroids to form localized sets. The outputs are groups of point sets of size $N_1 \times K \times 3$, where each group corresponds to a local region and K is the number of points in the neighborhood of centroid points. PointNet operation is a set function $f : \mathcal{W} \to \mathbb{R}$ to encode local region patterns $\{w_1, w_2, \cdots, w_{N_1}\}$ into feature vectors

$$f(w_1, w_2, \cdots, w_{N_1}) = \beta\left(\underset{i=1,2,\cdots,N_1}{\text{MAX}} \{\alpha(w_i)\}\right), \tag{2}$$

where α and β are MLP networks; MAX is a max pooling operation to aggregate the whole point set.

The environmental point cloud data obtained from the camera are denoted as p_e, and the point cloud data of the robot arm are denoted as p_r. These data are then fed into the refined PointNet++. Specifically, in the first abstraction layer, an iterative furthest point sampling procedure is deployed to generate a set of 1 024 points. Subsequently, we conduct a grouping query within a 10 cm radius, limiting the group size to a maximum of 256 points. This is followed by the PointNet structure, composed of three MLP layers with sizes of 32, 64 and 64, respectively.

The second abstraction layer operates at a lower resolution, employing a furthest point sampling technique to select 256 representative points. Grouping is then performed within a 30 cm radius, with each group containing up to 256 points. The corresponding PointNet architecture consists of layers with sizes of 64, 128 and 128, respectively.

In the third abstraction layer, furthest point sampling is omitted, and all points are grouped together. The subsequent PointNet component is composed of layers with sizes of 256, 512 and 512, respectively. Following the abstraction layers, we address the fusion of point cloud data related to the robotic arm's initial state and environment. For the robotic arm's point cloud data, we employ fully connected layer comprising 512 and 128 neurons, respectively. For the environmental information, our approach leverages fully connected layer with sizes of 512 and 256 neurons. Through the refined PointNet++ network, we obtain the environment point feature p'_e and the robot point feature p'_r.

Overall, the first layer's accurate analysis of local features and subsequent two layers' analysis on global features collectively ensure efficient, comprehensive processing of point cloud data of robot and environment. Our design aims to achieve the balance of detail and scope in environmental feature extraction, thereby optimizing our system for motion planning tasks.

4.1.2 LSTM network

The incorporation of LSTM network in BRMPNet offers three distinct advantages tailored for dynamic usage contexts. Unlike traditional feedforward networks, LSTM is good at retaining relevant historical information, owing to the gating mechanism that alleviates vanishing gradient problems. This capability enables the recognition of long-term dependencies that can be pivotal for future motion planning. Additionally, the adaptability of LSTM to irregular data sampling frequencies enhances its robustness, a critical feature when processing sensor data with varying sampling rates. Also, the recursive nature of LSTM enables real-time adaptability by iteratively updating motion plans based on both current environmental conditions and historical planning data. This is a vital feature in dynamic settings where real-time decision-making is significant.

Our LSTM network produces planned path $\sigma(x) = \{x_0, x_1, \cdots, x_{T-1}, x_T\}$. Within the context of planning, vital information for each step is extracted from refined PointNet++. We treat the environmental point feature p'_{e} as the memory cell internal state c_t and the concatenated feature of x_t and the robot point feature p'_{r} as the hidden state h_t. To address the challenge of generating goal-oriented path sequences, we adopt an approach of incorporating the goal's location x_{start} or x_{goal} as an auxiliary input during each prediction step. This auxiliary input serves as a constant reminder to the network regarding its ultimate convergence point. The output is obtained by passing $h_{(t+1)}$ through a fully connected layer to yield $x_{(t+1)}$. Since our LSTM network's execution is bidirectional, the motion planning will be conducted in both forward (i.e., from the start point to the goal) and backward (i.e., from the goal to the start point) ways. This bidirectional planning process continues until the forward and backward path converge. It's important to note that Fig. 2 illustrates the points x_{start} or x_{goal} explicitly. When planning forward, x_{goal} is input. While planning backward, x_{start} is used instead. A detailed algorithmic description of this process will be presented in Sect. 4.3.

4.2 Model training

The BRMPNet model is trained using a collection of expert-generated planning results that span the configuration space. Utilizing imitation learning techniques, BRMPNet aims to approximate the behavior of the expert algorithm. For each planning case provided by the expert, the network's output is compared against the expert's output using a behavior cloning loss function. Each expert planning case can be represented as $\sigma(\hat{x}) = \{\hat{x}_0, \hat{x}_1, \cdots, \hat{x}_{T-1}, \hat{x}_T\}$, which is a list of waypoints in configuration space that can be connected to the start and goal positions to form a path. BRMPNet generates a sequence of outputs, denoted as $\sigma(x)$, based on input conditions that are consistent with those used in expert planning cases. The behavioral cloning (BC) loss from one demonstrated planning case can be interpreted as follows

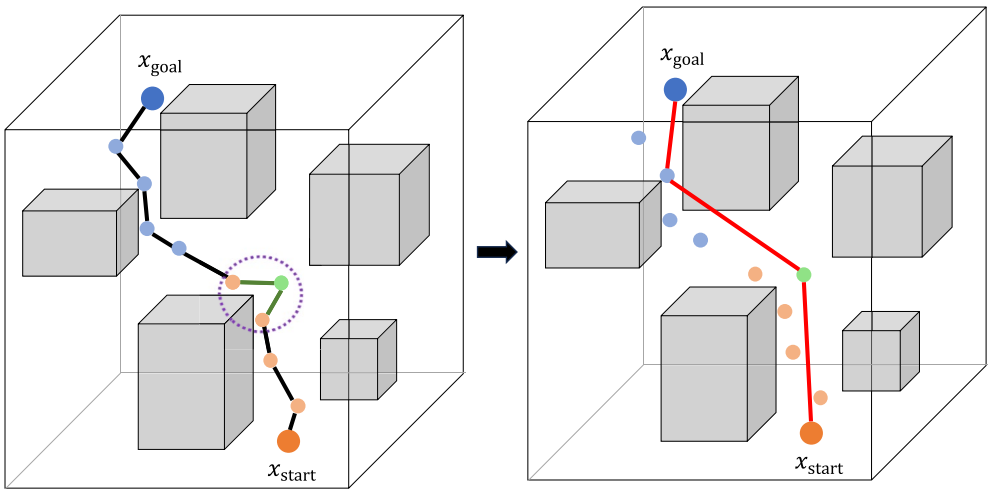

Fig. 3 Schematic representation of the replanning and rewiring processes in BRMPNet

$$L_{\mathrm{BC}} = \sum_{t=1}^{T-1} \left(\| \hat{x}_t - x_t \|_{L_1} + \| \hat{x}_t - x_t \|_{L_2} \right), \tag{3}$$

where L_1 and L_2 refer to the Manhattan and Euclidean norms, respectively. The L_1 norm is robust to outliers, as each component of the error contributes linearly to the total error. However, the L_1 norm is not smooth and lacks differentiability at zero. The squaring operation makes the L_2 norm sensitive to larger errors, effectively emphasizing them more in the total error calculation. However, the L_2 norm can be vulnerable to the influence of outliers. Using both norms can potentially leverage their respective benefits and create a more comprehensive and robust error measure.

4.3 Hybrid online execution

Algorithm 1 outlines the hybrid online execution logic of BRMPNet. Given a pre-trained BRMPNet, we focus on online path planning where both the start position x_{start} and the goal position x_{goal} are selected from $\mathcal{C}_{\mathrm{free}}$. The primary objective is to compute a path σ that connects x_{start} to x_{goal} while avoiding collisions with obstacles in the environment. To enhance the robustness of our path planning process, we implement a bi-directional path generation strategy. This strategy not only improves the feasibility and success rate of path planning with trivially additional computational cost, but also resolves key challenges in learning-based motion planning. Primarily, it reduces cumulative prediction error by decreasing the number of planning steps in each direction. Furthermore, in dynamic environments, the planning points generated in the backward direction serve as valuable references, aiding in the identification of alternative feasible paths and enhancing the system's adaptability to changing conditions. This strategy alternately extends paths from x_{start} or x_{goal}, allowing two sub-paths ($\sigma_{\mathrm{forward}}$ and $\sigma_{\mathrm{backward}}$) to grow towards each other. In Fig. 3, orange points indicate those generated during forward planning, while blue points denote those from backward planning. The algorithm terminates this phase once a feasible connection between these sub-paths is formed, resulting in a unified path σ (refer to Line 8 in Algorithm 1). If all sequentially generated points in σ are connectable (i.e., the path lies within a collision-free space), no further planning is necessary. However, if this is not the case, the next step involves assessing σ for beacon points, defined as consecutive and non-connectable waypoints in σ (see Line 14 in Algorithm 1). The discovery of beacon points triggers a replanning phase. As shown in Fig. 3a, the purple dotted circle encapsulates the area undergoing replanning, with green point generated during this phase. BRMPNet decomposes the given planning task into sub-problems and recursively solves them to find a path solution. If the replanning phase fails M times, a hybrid planning strategy is invoked (Line 18 shows $M = 1$). Then an RRT-based algorithm that uses the beacon points x_i and x_{i+1} as the start and goal points is employed, ensuring the algorithm's probabilistic completeness. Finally, after securing a collision-free path, a rewiring process is initiated to refine the path by eliminating redundant points. This process involves evaluating if connecting two non-adjacent points directly would reduce the overall path cost (Lines 10 and 21 in Algorithm 1). Figure 3b depicts the rewiring process with red lines representing the refined path after rewiring. In a word, our hybrid online execution mechanism not only maintains high computational efficiency but also ensures robust planning performance.

Input: $x_{\text{start}} \in \mathcal{C}_{\text{free}}, x_{\text{goal}} \in \mathcal{C}_{\text{free}}, T_{\max}$
Output: σ

$\sigma_{\text{forward}} \leftarrow \{x_{\text{start}}\}, \sigma_{\text{backward}} \leftarrow \{x_{\text{goal}}\}$
$\sigma \leftarrow \{\}$
direction $\leftarrow$ forward
for $t \in (0, T_{\max}]$ **do**
  $x_t \leftarrow \text{BRMPNet}(x_{\text{start}}, x_{\text{goal}}, \text{direction})$
  $\sigma_{\text{direction}} \leftarrow \sigma_{\text{direction}} \cup \{x_t\}$
  if FeasibleConnect($\sigma_{\text{forward}}, \sigma_{\text{backward}}$) **then**
    $\sigma \leftarrow \sigma_{\text{forward}} \cup \sigma_{\text{backward}}$
    if FeasiblePath(σ) **then**
      Rewire(σ)
      return σ
    else
      for $i \in [0, |\sigma|)$ **do**
        $x_i, x_{i+1} \leftarrow \text{IdentifyBeaconPoints}(\sigma)$
        $x_{\text{new}} \leftarrow \text{BRMPNet}(x_i, x_{i+1}, \text{direction})$
        if FeasibleConnect($x_i, x_{\text{new}}, x_{i+1}$) **then**
          $\sigma \leftarrow \sigma \cup \{x_{\text{new}}\}$
        else
          $\sigma \leftarrow \text{HybridPlanning}(x_i, x_{i+1}, \text{RRTPlanner})$
        $i \leftarrow i + 1$
      Rewire(σ)
      return σ
  direction $\leftarrow$ Opposite()
  $t = t + 1$

4.4 Analysis of the probabilistic completeness for BRMPNet

In this section, we provide proof of the probabilistic completeness for BRMPNet.

Theorem (Probabilistic completeness) BRMPNet attains probabilistic completeness for motion planning as the number of planning attempts goes to infinity.

Proof: In the planning phase, BRMPNet initially identifies rough solutions denoted as σ, which may encompass beacon points characterized by being unconnectable and consecutive (see Fig. 3a). Our proposed algorithm addresses the connectivity of these beacon points through replanning or hybrid planning procedure in online execution. During the replanning procedure, we subject the identified beacon states to neural replanning over a fixed number of M iterations, thereby iteratively refining the solutions represented by σ. In instances where our iterative neural replanning fails to yield a solution, we resort to employing a hybrid planning which utilizes the RRT-based algorithm to establish connections between any remaining unconnected beacon points. Thus, BRMPNet exhibits probabilistic completeness of the RRT-based planner. According to the probabilistic completeness of RRT [5], there exists a constant $a > 0$ that

$$\liminf_{n\to\infty} P\left(\sigma_n^{\text{BRMPNet}} \neq \emptyset\right) > \liminf_{n\to\infty} (1 - e^{-an}) = 1, \tag{4}$$

where $\sigma_n^{\text{BRMPNet}} \subset \sigma_{\text{feasible}}$ is the set of feasible paths found by BRMPNet planner. The probabilistic completeness proof for the BRMPNet algorithm is complete.

5 Experiments

5.1 Experiment settings

To examine the efficacy and efficiency of the proposed BRMPNet, a series of incremental experiments are designed to simulate common robotic operations in smart manufacturing. Experiment 1 performs the path planning within 2D and 3D environments, which represent the regular working environments for mobile robots such as AGVs or UAVs (unmanned aerial vehicles) in a simplified manner. This experiment aims to test the algorithmic performance in low-dimensional space by eliminating complexities introduced by robot dynamics. Building on this, Experiment 2 introduces a desktop multi-joint robotic arm navigating through static obstacles, a scenario reflective of common material-handling or assembly tasks in smart manufacturing. This experiment enables us to assess BRMPNet's performance in higher-dimensional configuration spaces and its real-world applicability for industrial robots. Experiment 3 escalates the complexity by focusing on dynamic obstacle-avoidance within the same desktop robotic arm setup in Experiment 2. This serves to evaluate the algorithm's robustness and adaptability to variable environments. Experiment 4 involves a collaborative assembly task between a hybrid robot (i.e., a mobile robotic platform equipped with a robotic arm) and a desktop robotic arm, mimicking the multi-robot cooperative scenarios frequently seen in smart manufacturing. This test is designed to evaluate BRMPNet's scalability and efficiency in multi-robot dynamic environments. Collectively, these four experiments provide a comprehensive assessment of BRMPNet in an incremental way, starting from basic to complex, real-world operations commonly encountered in generic robotic platforms. These incremental tests enable a thorough evaluation of the proposed algorithm as well as closely align with the procedures that a novel motion planning algorithm will take from laboratory testing to practical deployment in industry.

We compare the performance of the proposed BRMPNet approach against several well-established motion planning algorithms, namely RRT [5], RRT-Connect [35], batch-informed RRT (BIT) [36] and MPNet [9]. MPNet utilizes an end-to-end learning architecture, employing contractive autoencoders for environmental encoding and MLP for the planning module. For the BIT algorithm, we configure it to operate with a batch size comprising 50 sampling points. All algorithms are implemented in Python, utilizing a uniform planning framework, and are tested on a computer with an Intel i7-10300 CPU and 32 GB RAM.

5.2 Experiment results

5.2.1 2D and 3D scenarios with mass-point type robot

Experiment 1 aims to assess the fundamental capabilities of BRMPNet in simplified 2D and 3D environments. The 2D workspace is a square area with sides extending from negative 20 to positive 20 units along both axes. The 3D workspace is a cube with sides ranging from negative 20 to positive 20 units along the X, Y and Z axes. In the 2D setup, we introduce ten identically-sized rectangular obstacles, while the 3D setup features ten randomly-sized cuboid obstacles. These obstacles are randomly placed to ensure no complete overlap exists, and are converted to point cloud data for training. We construct training datasets for both 2D and 3D environments, encompassing a total of 100 distinct training scenarios. Each scenario comprises 3 000 pairs of randomly generated initial and goal states. The expert paths for training datasets are derived using RRT algorithm implemented in Python. An additional 10 test scenarios, distinct from the training set, are created to ensure experimental fairness. In the comparative analysis, RRT, RRT-Connect, and BIT algorithms are used directly without the need for pre-training. In contrast, BRMPNet and MPNet are trained on the same dataset. Exemplary planning results of BRMPNet are presented in Fig. 4. Figures 4a and c show the paths generated by BRMPNet in an end-to-end manner. As depicted in Figs. 4b and d, BRMPNet serves as a preprocessing module for RRT algorithm by generating informed sampling points that direct RRT in forming a path.

Figure 5 compares the planning time (mean time $\pm$ standard deviation) of different algorithms for achieving initial solutions across 1 000 test cases in 2D and 3D scenarios. In both 2D and 3D scenarios, the planning time of traditional sampling-based algorithms (RRT, RRT-Connect, and BIT) are consistently higher than those of learning-based algorithms (MPNet and BRMPNet). This underscores the efficiency benefits of learning-based methods in motion planning. The proposed BRMPNet algorithm outperforms all others in terms of planning time. In the 2D scenario, BRMPNet achieves an initial solution in just 0.19 ± 0.07 s, which is markedly faster than the top-performing sampling-based algorithm, BIT, which requires 0.93 ± 0.21 s. This indicates that BRMPNet operates at approximately 20% of the planning time of BIT in this context. The trend is consistent in the 3D scenario, where BRMPNet demands only 0.37 ± 0.09 s, showcasing its scalability across different complexities. Furthermore, the notably lower standard

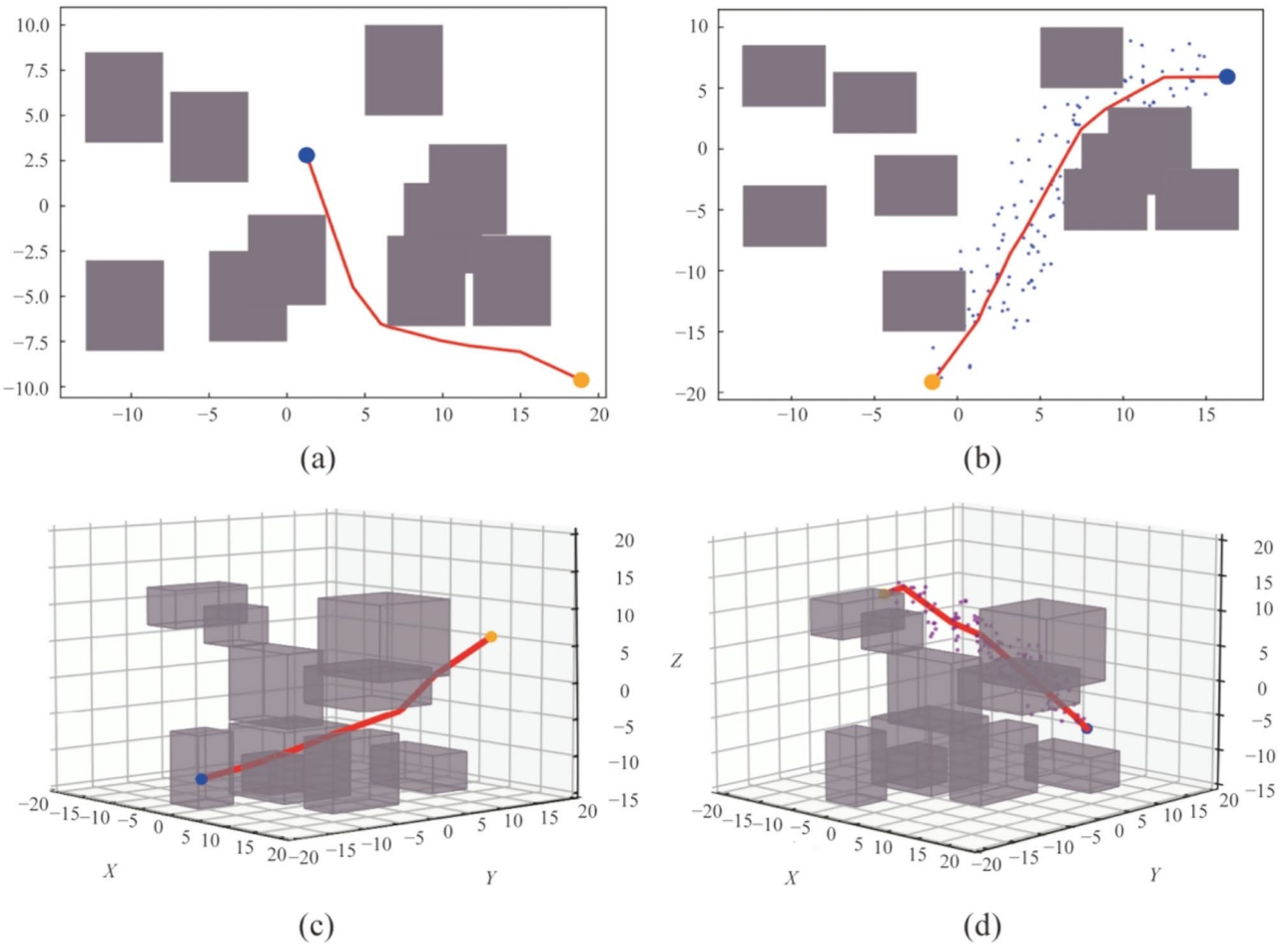

Fig. 4 Exemplary planning results of BRMPNet in 2D and 3D environments

deviation associated with BRMPNet compared to other algorithms indicates its robustness and reliability across a variety of test cases.

Compared to MPNet, the other learning-based algorithm, BRMPNet consistently demonstrates superior performance. For example, in 2D scenarios, BRMPNet achieves approximately twice the speed of MPNet. This efficiency underscores the impact of BRMPNet's advanced architectural design, which facilitates the generation of highly informed points, crucial for efficient pathfinding.

The modified version of BRMPNet, named as BRMPNet(S), is integrated as a preprocessing module within the RRT algorithm to act as an informed sampling function. According to Figs. 4c and d, the informed sampling points generated by BRMPNet play a crucial role in obtaining the initial solution. In both 2D and 3D scenarios, the integration of BRMPNet(S) with RRT reduces the computation time to approximately one-third of the original RRT: 0.82 ± 0.19 s from 2.10 ± 0.31 s in 2D, and 1.28 ± 0.23 s from 3.76 ± 0.40 s in 3D. This demonstrates a roughly triple speed-up, signifying that BRMPNet(S) enhances RRT's performance significantly. This integration showcases the flexibility and versatility of BRMPNet to seamlessly switch between end-to-end and module replacement strategies.

5.2.2 Static obstacle-avoidance with desktop robotic arm

In Experiment 2, a desktop multi-joint robotic arm navigates through static obstacles, representing common material-handling or assembly tasks in smart manufacturing. As illustrated in Fig. 6a, distinct obstacles: cubes, cones, and labeling machine are incorporated in the Gazebo simulation environment. Multiple Kinect cameras, strategically positioned to minimize occlusions and maximize the field of view, are used to capture point cloud data from the environment setup (see Fig. 6b). The point cloud data are then merged and refined through subsequent point cloud fusion operations. BRMPNet utilizes point cloud information for planning which can be validated (see Fig. 6c) and sent to the physical robot arm (see Fig. 6d). We design ten training

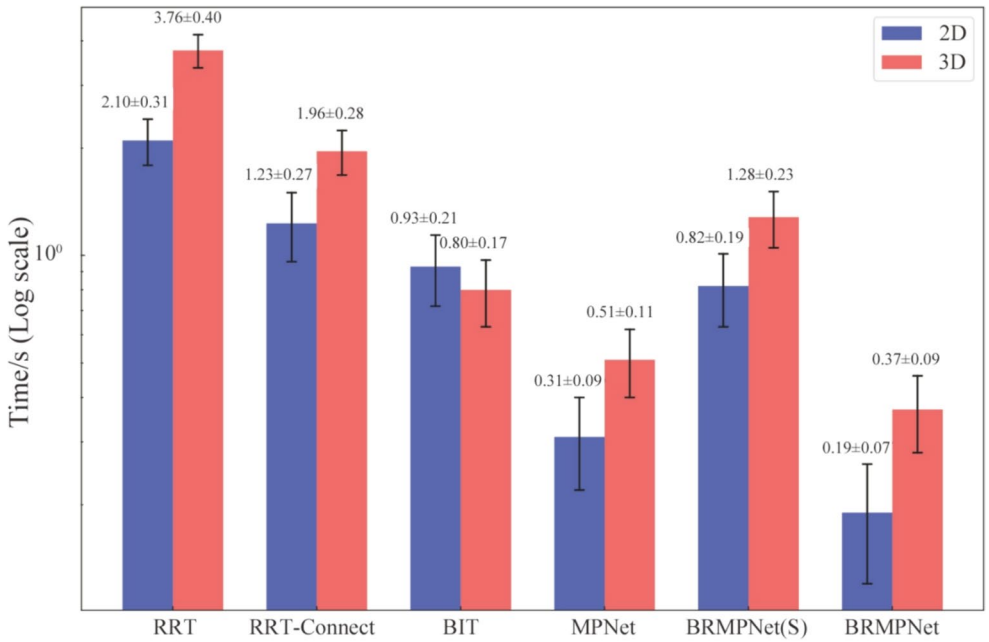

Fig. 5 Planning time of various algorithms to get initial solutions for 1 000 test cases in both 2D and 3D environments

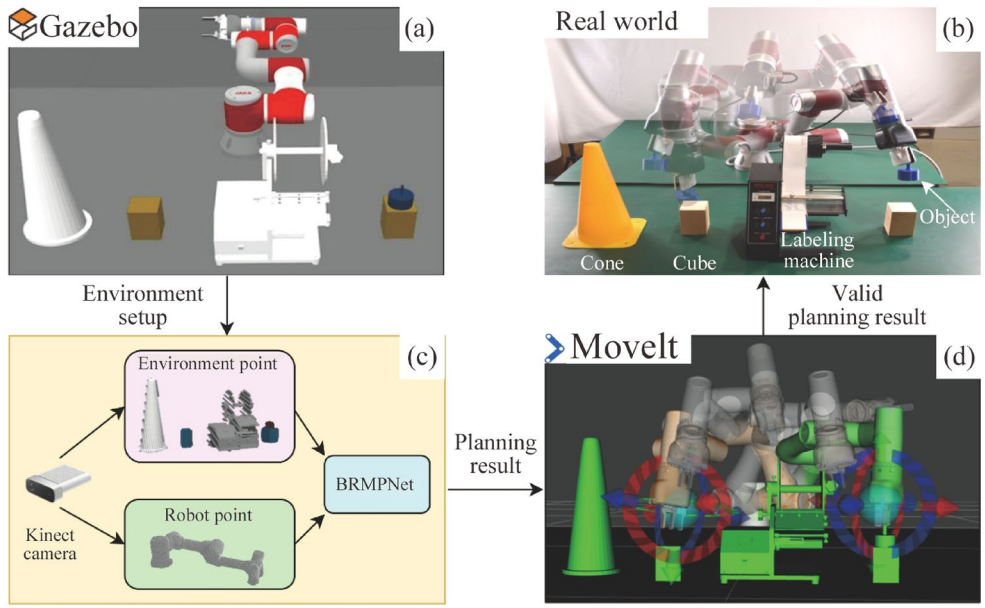

Fig. 6 Experiment 2 setup and the information flow of BRMPNet

scenarios by placing various numbers and combinations of obstacles. This diverse setup can represent real-world environment as well as introduce varying complexities for the navigation algorithms. To ensure comprehensive spatial coverage and eliminate potential bias, each of these scenarios include a training set comprised of 3 000 randomized pairs of start and goal points. The expert paths needed for our model training are obtained from the RRT algorithm within the MoveIt robotics platform, utilizing the open motion planning library (OMPL) plugin [37] . It's crucial to note that the test scenario in Fig. 6 is different from the training scenarios to ensure the fairness of our experiment. We conduct 100 trials in which the robot arm places the object (i.e., the blue item in Fig. 6d) on top of one cube and then moves it to the top of another cube in the test scenario.

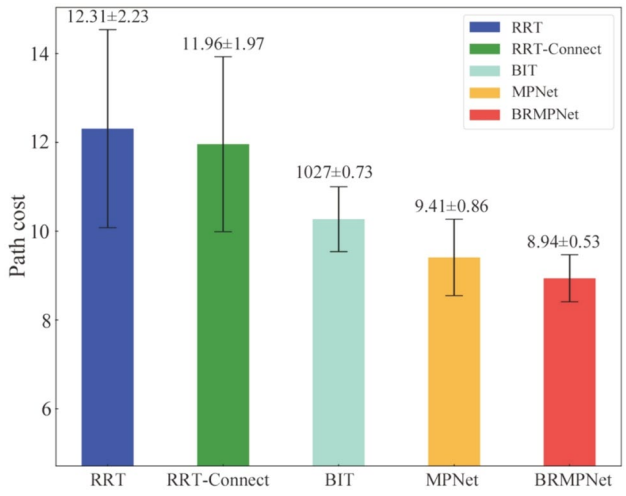

Fig. 7 Path cost comparison for different algorithms to obtain initial solutions within one second

Table 1 Success rate of different algorithms to obtain initial solutions within one second (%)

	RRT	RRT-Connect	BIT	MPNet	BRMPNet
Success rate	77	85	87	83	91

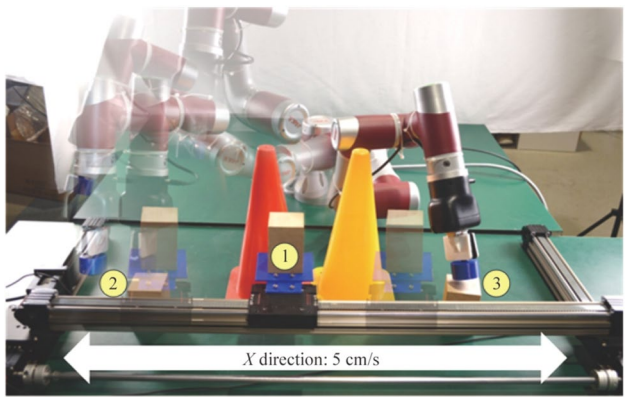

Fig. 8 Experiment 3 setup and the trajectory of the robotic arm in motion

Table 2 Success rate for different algorithms to complete tasks (%)

	RRT	RRT-Connect	BIT	MPNet	BRMPNet
Success rate	–	–	–	80	93

Figure 7 depicts the path cost (mean cost ± standard deviation) of various algorithms for obtaining an initial solution within the planning time limit of one second. The path cost is a quantitative metric to encapsulate the optimality and feasibility of trajectories generated by motion planning algorithms. A lower path cost usually signifies a more direct and efficient trajectory, while higher values might indicate circuitous routes or potential inefficiencies in the planning process. BRMPNet exhibits the lowest path cost (8.94 ± 0.53), underscoring its prowess in planning near-optimal paths in rapid time. Traditional sampling-based algorithms show higher path costs, with RRT topping the list at 12.31 ± 2.23. MPNet gets a path cost of 9.41 ± 0.86, while commendable, still falls short when compared with BRMPNet. The success rate quantifies the consistency and reliability of an algorithm in successfully finding a path within the specified time limit. Table 1 presents the success rates of various algorithms for obtaining initial solutions within one second. BRMPNet again shines with a 91% success rate, outperforming other methods. Overall, BRMPNet not only yields a better success rate but also achieves a lower path cost in Experiment 2, highlighting its superiority in motion planning.

5.2.3 Dynamic obstacle-avoidance with desktop robotic arm

In Experiment 3, we increase the task difficulty by adding dynamic obstacle into the desktop robotic arm environment. This experiment serves as a stringent test of the algorithm's adaptability in changing environmental intricacies. The BRMPNet model parameter utilized is trained with the same dataset from Experiment 2. A pivotal distinction in this experiment lies in the handling of point cloud data. We leverage a depth camera to capture the point cloud data of a moving cube obstacle. As depicted in Fig. 8, the cube 1 obstacle is strategically situated on a sliding platform that traverses in the X-direction at a speed of 5 cm/s. To account for the dynamic obstacle, the BRMPNet operates synchronously with the point cloud fusion, outputting each planning step at a consistent frequency of 5 Hz. One hundred tests are conducted, moving the blue object from cube 2 to cube 3 in this scenario. Table 2 shows the planning success rate of the learning-based approach in the dynamic scenario. Traditional sampling-based algorithms like RRT, RRT-Connect, and BIT are not effective in dealing with moving obstacles, as they do not predict or respond to changes in obstacle positions without specific reconfigurations. Consequently, sampling-based algorithms are not involved in algorithmic effectiveness comparisons.

As shown in Table 2, BRMPNet, with a success rate of 93%, stands out as the top performer in dynamic scenario. The other learning-based method MPNet achieves an 80% success rate. The 13% advantage underscores the BRMPNet in terms of adaptability and robustness. While BRMPNet achieves a success rate of 93%, it is also important

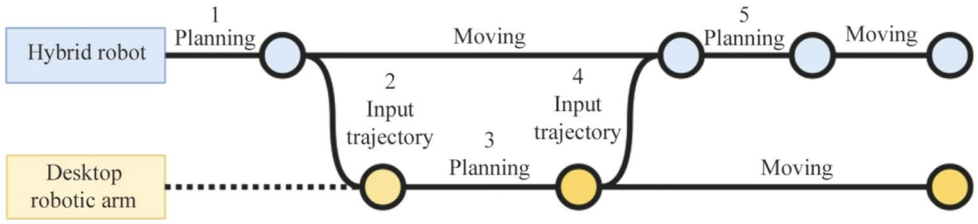

Fig. 9 Procedure of the collaborative assembly with dual robotic arms

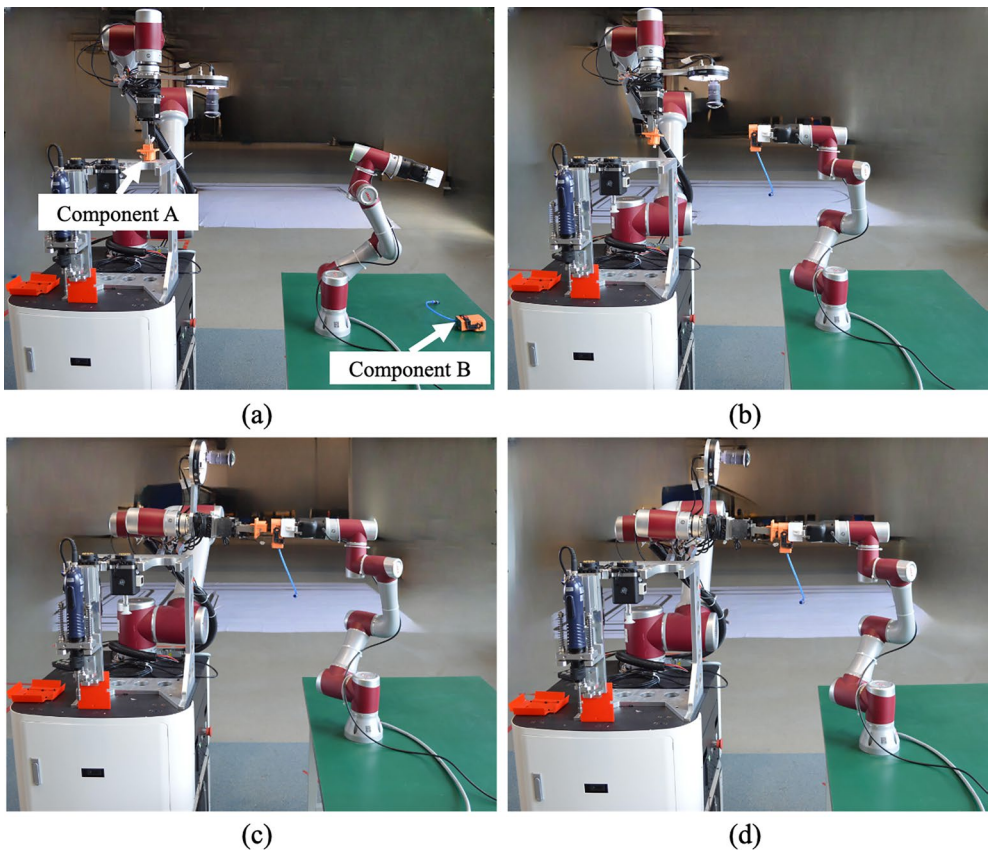

Fig. 10 Illustration of the dual-arm coordinated assembly task robotic arms

to analyze the factors preventing a 100% success rate. One possible reason is that the current algorithm considers scene modifications to update point cloud solely after the robot reaches its previous planning step. Additionally, real-world point cloud data acquisition is rarely perfect. The presence of noise, or occlusions can lead to imperfect scene representations. These imperfections, in turn, can introduce discrepancies in the planning process, sometimes causing the robot to misinterpret or misjudge obstacles.

In summary, BRMPNet algorithm excels in dynamic robotic planning, outperforming its counterparts. Its adaptability and real-time responsiveness, especially in environments with moving obstacles, sets it apart.

5.2.4 Dual-arm coordinated assembly with hybrid robot and desktop robot arm

Experiment 4 explores collaborative assembly by integrating a hybrid robot, which combines a mobile robot platform with a robotic arm, and a desktop robotic arm. This configuration reflects common multi-robot collaborative scenarios seen in smart manufacturing. Our main goal is to investigate the

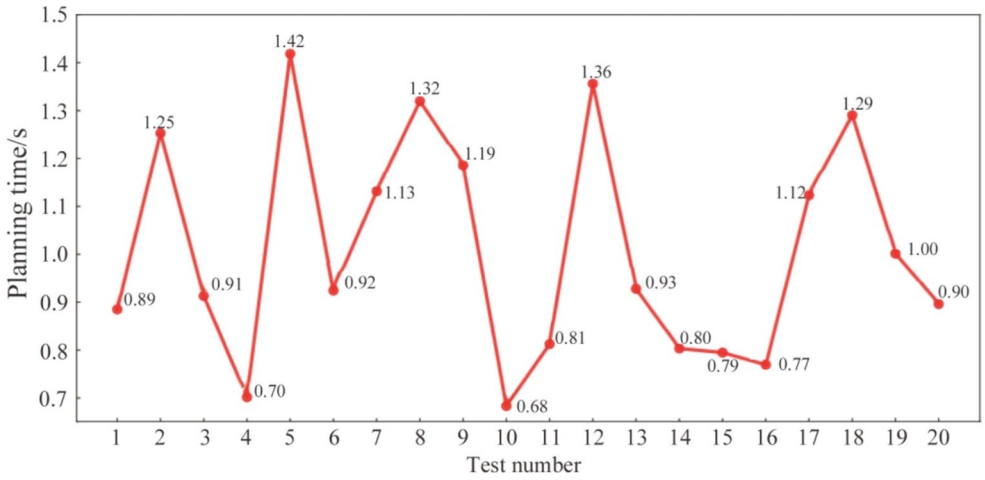

Fig. 11 Planning time of BRMPNet for 20 tests in Experiment 4

scalability and efficiency of BRMPNet in a dynamic multi-robot scenario. Figures 9 and 10 illustrate the procedure of the collaborative assembly of the two robotic arms with schematic diagrams and site photos, respectively. In the initial stage, the hybrid robot coordinates the grasp planning for component A (see Fig. 10a). Subsequently, the desktop robotic arm plays the supporting role by grasping and holding the component B (see Fig. 10b). Finally, the hybrid robot attempts to assemble component A with component B (see Figs. 10c, d).

As shown in Fig. 11, during the collaborative assembly task, we conduct 20 tests of BRMPNet and find that it meets the rigorous requirements of real-time applications with an average planning time of one second. This rapid response time is indicative of BRMPNet's inherent adaptability in dynamically changing environments. Such proficiency speaks to the algorithm's ability to process complex information efficiently as making real-time decisions is critical in high-paced, intricate manufacturing scenarios. Furthermore, BRMPNet's proven scalability and efficiency in cooperative frameworks render it a suitable choice for further explorations and applications in smart manufacturing.

6 Conclusions

The contemporary shift towards Industry 4.0 demands continuous evolutions in robotic motion planning. As traditional motion planning approaches grapple with the rising complexities of working environments in smart manufacturing, the development for alternative solutions becomes imperative. We introduce the BRMPNet, a novel architecture that employs imitation-based learning technique to achieve highly efficient real-time motion planning. The integration of refined PointNet++ enables fully utilizing point cloud information from depth sensors. The design of bi-directional recurrent neural networks aids in capturing the dynamic temporal dependencies of planning results. Furthermore, BRMPNet's compatibility with traditional sampling-based algorithms not only amplifies its versatility but also fortifies its reliability. Experiment results consistently underscore BRMPNet's superior capabilities against other state-of-the-art algorithms across various configurations of generic robotic platforms, both in solution quality and computational efficiency. This offers the compelling evidence of the transformative potential of BRMPNet in addressing the multifaceted challenges of robotic applications in smart manufacturing.

While our work establishes BRMPNet as a robust framework for robot motion planning, there are still several promising directions for future research. One key direction involves investigating methods for swiftly accommodating fine-tuned pre-trained BRMPNet models to new applications. This could further reduce the training time and computational load, streamlining the deployment process. Another possible direction is the integration of human-robot interaction within the BRMPNet framework. As smart manufacturing increasingly necessitates seamless collaboration between human and robots, improving BRMPNet for safe and effective human-robot collaborative working space will become critical. Collectively, these future research directions will bolster the capabilities of BRMPNet and align it more closely with the emerging requirements of Industry 4.0.

Acknowledgment The authors would like to acknowledge the financial support from National Key R&D Program of China (Grant No. 2022YFB4702400).

References

1. Strandhagen JW, Alfnes E, Strandhagen JO et al (2017) The fit of Industry 4.0 applications in manufacturing logistics: a multiple case study. Adv Manuf 5:344–358
2. Xie DJ, Zeng LD, Xu Z et al (2023) Base position planning of mobile manipulators for assembly tasks in construction environments. Adv Manuf 11:93–110
3. He B, Bai KJ (2021) Digital twin-based sustainable intelligent manufacturing: a review. Adv Manuf 9:1–21
4. Zhang Z, He R, Yang K (2022) A bioinspired path planning approach for mobile robots based on improved sparrow search algorithm. Adv Manuf 10:114–130
5. LaValle SM (1998) Rapidly-exploring random trees: a new tool for path planning. Tech Rep 98:11
6. Kavraki LE, Svestka P, Latombe JC et al (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom 12:566–580
7. Liao J, Huang F, Chen Z et al (2019) Optimization-based motion planning of mobile manipulator with high degree of kinematic redundancy. Int J Intell Robot Appl 3:115–130
8. Ichter B, Pavone M (2019) Robot motion planning in learned latent spaces. IEEE Robot Autom Lett 4:2407–2414
9. Qureshi AH, Miao Y, Simeonov A et al (2021) Motion planning networks: bridging the gap between learning-based and classical motion planners. IEEE Trans Robot 37:48–66
10. Kumar R, Mandalika A, Choudhury S et al (2019) LEGO: leveraging experience in roadmap generation for sampling-based planning. In: 2019 IEEE/RSJ International conference on intelligent robots and systems (IROS), Minneapolis, USA, pp 1488–1495
11. Wang J, Zhang T, Ma N et al (2021) A survey of learning-based robot motion planning. IET Cyber-Syst Robot 3:302–314
12. Pfeiffer M, Schaeuble M, Nieto J et al (2017) From perception to decision: a data-driven approach to end-to-end motion planning for autonomous ground robots. In: 2017 IEEE International conference on robotics and automation (ICRA), Singapore, pp 1527–1533
13. Hamandi M, D'Arcy M, Fazli P (2019) DeepMoTIon: learning to navigate like humans. In 2019 IEEE international conference on robot and human interactive communication (RO-MAN), New Delhi, India, pp 1–7
14. Bency MJ, Qureshi AH, Yip MC (2019) Neural path planning: fixed time, near-optimal path generation via oracle imitation. In: 2019 IEEE/RSJ International conference on intelligent robots and systems (IROS), Macau, China, pp 3965–3972
15. Fishman A, Murali A, Eppner C et al (2022) Motion policy networks. In: Proceedings of the 6th conference on robot learning (CoRL), Auckland, New Zealand, pp 967–977
16. Kurutach T, Tamar A, Yang G et al (2018) Learning plannable representations with causal InfoGAN. In: Proceedings of the 32nd international conference on neural information processing systems, Red Hook, USA, pp 8747–8758
17. Huh J, Isler V, Lee DD (2021) Cost-to-go function generating networks for high dimensional motion planning. In: 2021 IEEE international conference on robotics and automation (ICRA), Barcelona, Spain, pp 8480–8486
18. Oh J, Singh S, Lee H (2017) Value prediction network. In: Proceedings of the 31st conference on neural information processing systems (NIPS), Long Beach, USA, pp 6118–6128
19. Al-Hilo A, Samir M, Assi C et al (2021) UAV-assisted content delivery in intelligent transportation systems-joint trajectory planning and cache management. IEEE Trans Intell Transp Syst 22:5155–5167
20. Strudel R, Pinel RG, Carpentier J et al (2021) Learning obstacle representations for neural motion planning. In: Proceedings of the 2020 conference on robot learning (CoRL), pp 355–364
21. Khan A, Ribeiro A, Kumar V et al (2020) Graph neural networks for motion planning. arXiv 2006.06248
22. Wang J, Chi W, Li C et al (2020) Neural RRT*: learning-based optimal path planning. IEEE Trans Autom Sci Eng 17:1748–1758
23. Ying KC, Pourhejazy P, Cheng CY et al (2021) Deep learning-based optimization for motion planning of dual-arm assembly robots. Comput Ind Eng 160:107603
24. Qureshi AH, Yip MC (2018) Deeply informed neural sampling for robot motion planning. In: 2018 IEEE/RSJ International conference on intelligent robots and systems (IROS), Madrid, Spain, pp 6582–6588
25. Elhafsi A, Ivanovic B, Janson L et al (2020) Map-predictive motion planning in unknown environments. In 2020 IEEE international conference on robotics and automation (ICRA), Paris, France, pp 8552–8558
26. Kim S, An B (2020) Learning heuristic a: efficient graph search using neural network. In 2020 IEEE international conference on robotics and automation (ICRA), Paris, France, pp 9542–9547
27. Guzzi J, Chavez-Garcia RO, Nava M et al (2020) Path planning with local motion estimations. IEEE Robot Autom Lett 5:2586–2593
28. Chase KJ, Ichter B, Bandari M et al (2020) Neural collision clearance estimator for batched motion planning. In: 2020 International workshop on the algorithmic foundations of robotics (WAFR), Oulu, Finland, pp 73–89
29. Zhang C, Huh J, Lee DD (2018) Learning implicit sampling distributions for motion planning. In: 2018 IEEE/RSJ International conference on intelligent robots and systems (IROS), Madrid, Spain, pp 3654–3661
30. Tran T, Denny J, Ekenna C (2020) Predicting sample collision with neural networks. arXiv 2006.16868
31. Yu C, Gao S (2021) Reducing collision checking for sampling-based motion planning using graph neural networks. Adv Neural Inf Process Syst 34:4274–4289
32. Qi CR, Yi L, Su H et al (2017) PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the 31st international conference on neural information processing systems (NIPS), Red Hook, USA, pp 5105–5114
33. Zhou Y, Tuzel O (2017) VoxelNet: end-to-end learning for point cloud based 3d object detection. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), Salt Lake City, UT, USA, pp 4490–4499
34. Charles RQ, Su H, Kaichun M et al (2017) PointNet: deep learning on point sets for 3D classification and segmentation. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), Honolulu, HI, USA, pp 77–85
35. Kuffner JJ, LaValle SM (2000) RRT-connect: an efficient approach to single-query path planning. In: Proceedings of 2000 IEEE international conference on robotics and automation (ICRA), San Francisco, USA, pp 995–1001
36. Gammell JD, Barfoot TD, Srinivasa SS (2020) Batch informed trees (BIT*): informed asymptotically optimal anytime search. Int J Rob Res 39:543–567
37. Sucan IA, Moll M, Kavraki LE (2012) The open motion planning library. IEEE Robot Autom Mag 19:72–82

Bo-Han Feng is currently a Ph.D. student at University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, China. He received B.S. degree in Computer Science and Engineering from Dalian University of Technology in 2020. His research focuses on robot motion planning and human-robot collaboration algorithms in smart manufacturing.

Bo-Yan Li is currently a M.S. student at University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, China. He received B.S. degree in Mechanical Engineering from Shanghai Jiao Tong University. His research interests include 6D pose estimation, motion planning and robotics.

Xin-Ting Jiang is currently an undergraduate student in University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University. His research interests are computer vision algorithms and robotics.

Qi Zhou is currently a Ph.D. student at University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, China. He received B.S. degree in Mechanical and Electronic Engineering from Shandong University, China, in 2021. His research topics include robotics and reinforcement learning.

You-Yi Bi is currently an associate professor at University of Michigan – Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, China. He obtained his Ph.D. in Mechanical Engineering from Purdue University in 2017. His research interests include data-driven design, AI for robotics and smart manufacturing.