https://doi.org/10.1016/j.rcim.2022.102518

Neural Reactive Path Planning with Riemannian Motion Policies for Robotic Silicone Sealing

Peng Zhou[a], Pai Zheng[b], Jiaming Qi[c], Chengxi Li[b], Anqing Duan[a], Maggie Xu[a], Victor Wu[a], David Navarro-Alarcon[a,*]

[a]*Department of Mechanical Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong SAR, China*
[b]*Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong SAR, China*
[c]*Astronautics School, Harbin Institute of Technology, Heilongjiang, China,*

Abstract

Due to its excellent chemical and mechanical properties, silicone sealing has been widely used in many industries. Currently, the majority of these sealing tasks are performed by human workers. Hence, they are susceptible to labor shortage problems. The use of vision-guided robotic systems is a feasible alternative to automate these types of repetitive and tedious manipulation tasks. In this paper, we present the development of a new method to automate silicone sealing with robotic manipulators. To this end, we propose a novel neural path planning framework that leverages fractional-order differentiation for robust seam detection with vision and a Riemannian motion policy for effectively learning the manipulation of a sealing gun. Optimal control commands can be computed analytically by designing a deep neural network that predicts the acceleration and associated Riemannian metric of the sealing gun from feedback signals. The performance of our new methodology is experimentally validated with a robotic platform conducting multiple silicone sealing tasks in unstructured situations. The reported results demonstrate that compared with directly predicting the control commands, our neural path planner achieves a more generalizable performance on unseen workpieces and is more robust to human/environment disturbances.

Keywords: Robotic sealing, reactive path planning, Riemannian motion policy, neural path planning, seam detection.

1. Introduction

Silicone sealing is a common technique to join two surfaces/materials with special silicone sealant as the caulk [1]. After producing reactions with atmospheric moisture, the heavy-duty and gel-like adhesive offers exceptional adhesion performance, as well as resistance to large temperature variations, corrosive fluids, and even fire. Due to its excellent chemical and mechanical properties, silicone sealing has been extensively used in many industries, ranging from general construction and manufacturing [2, 3, 4] to home repair and device maintenance [5, 6, 7]. Up to this day, the majority of these sealing tasks are performed by human workers, hence, it is susceptible to labor shortage problems. The use of vision-guided robotic systems is a feasible alternative to automate these types of repetitive and tedious manipulation tasks [8]. However, this automation problem has not been sufficiently studied in the literature.

Creating a robotic system capable of autonomous sealing in complex and dynamic environments is a challenging problem. Unlike the system in static environments, less attention has been given to autonomous sealing in dynamic environments, which, besides stationary obstacles, also needs to deal with moving obstacles that must be avoided as well. For example, home repair and device maintenance always need to be considered under dynamic environments. As shown in Fig. 1, when sealing for fixing window leaks in a domestic environment, traditional robot control approaches based on e.g., "teach and playback" mode [9] or offline ad-hoc programming [10], are unpractical, as these control approaches lack adaptation to accommodate for changes in the environment. A viable solution is to exploit visual information [11, 12] to detect the sealing seam/joints, plan the path, and correct for deviations in real-time [13, 14, 15]. Therefore, the key to an intelligent sealing robot that can operate in dynamic environments lies in an efficient design of vision-based path planning and control [16, 17].

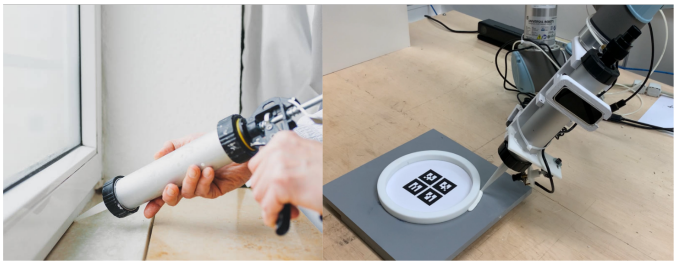

Fig. 1: (Left) Illustration of silicone sealing performed by a human expert and (right) by a robot arm.

Previous vision-based path planning approaches for sealing and welding applications are mainly divided into two working paradigms. The first employs real-time path planning on the basis of defining Regions of Interest (ROI) [18] and extracting seam characteristics [19, 20, 21]. The idea is to iteratively call the local path planning function with the input of real-time

⋆This work was supported by the Chinese National Engineering Research Centre for Steel Construction Hong Kong Branch (grant BBV8).

*Corresponding author e-mail: dna@ieee.org

updated visual clues in the ROIs. By measuring the changes in the detected seam characteristics, the robot is able to react promptly to environmental changes. However, this paradigm has been typically validated on simple workpieces that required motions in a straight two-dimensional segment [22]; This limits its applicability to general free-form grooves (e.g., with curved edges). The second paradigm relies on the availability of complete 3D geometric information of grooves and seams to implement offline path planning algorithms [23, 24]. By collecting partial visual/depth information, it computes and describes the groove/seams with a model, e.g., based on point clouds [25, 26] or RGB-D images [27, 28], then conducts 3D feature extraction to obtain the offline path for the sealing/welding task. The latter approach can handle complex grooves/seams with various workpieces, however, it depends on the availability of such precise geometry mode, which might be difficult to obtain in practice; Furthermore, as the planned path with this model-based approach typically assumes a static working environment, therefore, it is not robust to uncertain models (e.g., arising from the typical noise of the 3D vision sensors or kinematic errors of the sealing tool) [29, 30]. Recent efforts have been made consistently to implement a neural network to map visual input signals to path planning commands since [31]. Thanks to the development of deep learning, recent studies [32, 33, 34] have achieved impressive path planning performance with neural policies. However, the majority of previous efforts leverage imitation learning to map visual inputs to control commands in a supervised manner. According to this viewpoint, our method is likewise supervised, and the expert policy is a Riemannian Motion Policy (RMP) [35] controller. But unlike most previous research, our controller predicts a representation that directly describes the sealing gun and environmental dynamics using a neural network.

In this paper, we present a new autonomous sealing system capable of online path planning and control in complex dynamic environments; By using real-time feedback, our method avoids acquiring accurate 3D models of the workpiece. To this end, a fractional order differentiation-based edge detection is implemented to robustly extract the sealing seam from low Signal-to-noise Ratio (SNR) images (which are typically captured by a commodity consumable two-camera system). Subsequently, we introduce a novel framework for reactive path planning that leverages on RMPs and deep learning techniques. RMP models the interaction between the robot and the working environment as a unified representation, which is defined by an acceleration policy and its corresponding Riemannian metric. By utilizing the forward kinematics, optimal commands for the robot control can be generated computationally. This RMP technique is combined into the design of a neural path planning framework for robotic sealing tasks, as shown in Fig. 2. Compared to directly predicting the sealing control commands with neural networks, our neural network-based RMPs prediction is able to achieve more generalizable performance on unseen workpieces. Besides, by checking each predicted RMP, it is efficient to analyze and debug the desired sealing behavior, thus making the path planning approach in explainable manner.

To validate the proposed methodology, we developed an experimental system composed of a Universal Robots UR5 manipulator, a modified pneumatic sealing gun, and a real-time control system integrated into ROS (see Fig. 1 and Fig. 8). This system controls the airflow to the gun to deposit silicone sealant along the real-time planned path, which is computed from the RMPs; This motion policy adjusts the sealing velocity in an appropriate range and orientation for the detected workpieces. A detailed experimental study (with over 100 trials) is conducted to validate the performance of the automated sealing robot; Robustness to external disturbances is demonstrated with various experiments in dynamic situations. The conducted experimental results can be downloaded from: https://sites.google.com/view/reactive-sealing.

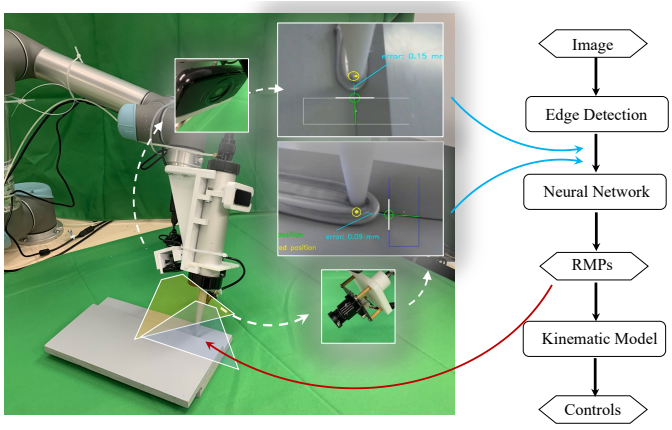

Fig. 2: Overview of the neural path planning framework with RMPs for automatic robotic sealing. The interaction between the sealing gun and the workpiece environment is modeled with neural RMPs. Typically, the interaction could be attractive (yellow), and the heading correction (green). The RMPs are predicted by training a neural network with the ROI features from two different camera views. The RMPs are then combined into the forward kinematics of the robotic sealing system to solve the control commands.

In summary, we present four key contributions in this paper:

- A novel two-camera path planning framework for robotic sealing;
- A fractional order differentiation seam detection method to robustly extract features in low-SNR images;
- A reactive path planning with RMPs to handle external disturbances in a dynamic environment;
- A detailed experimental validation of the proposed neural path planning framework with a robotic sealing platform.

The rest of the paper is organized as follows. Sect. 2 introduces the architecture of the proposed neural path planning system, together with key techniques applied, followed by Sect. 3 describing the system implementation details. The experimental results of the proposed system are presented in Sect. 4. Sect. 5 gives final conclusions

2. Methodology

The neural path planning framework for the robotic sealing system is presented in Fig. 2, where a novel image-based reactive autonomous sealing with an elaborately designed edge

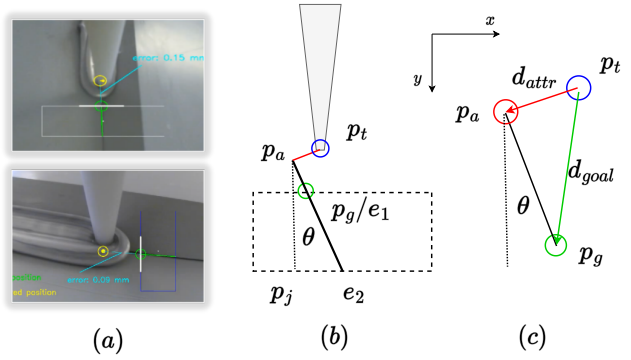

Fig. 3: Illustrations of the state representation for the sealing seam in the Kalman filter. (a) shows the real state representation of ROI features in front and side views of the robotic sealing task; (b) presents the conceptual representation of the ROI-related state variables, and (c) enlarges the attraction pattern between the sealing gun tip and the attraction point.

detection pipeline and the RMPs are proposed. It begins with Riemann-Liouville (R-L) fractional order derivative-based edge detection approach to extract the edge information in the ROIs from different camera perspectives. Subsequently, the Kalman filter is employed to further improve the precision of estimated ROI features. Then, a neural net is trained to predict RMPs, which model the interaction between the sealing gun tip and the workpiece into a unified robot geometry. This joint representation is typically defined by an acceleration policy with the corresponding Riemannian metric. By doing so, this RMP structure is combined into the design of our neural robotic sealing framework.

2.1. Robust Seam Detection Pipeline

To extract the edge features from two cheap commodity monocular camera sensors, the low SNR issue is unavoidable in a dynamic environment. Due to the limitations of the visual sensors, the resulting noise can decrease image quality and undermine the performance of following downstream sealing tasks. Therefore, it is of great importance to select an appropriate approach for contrast and texture enhancement to restore image features. Recently, fractional differential-order-based algorithms have demonstrated better results on contrast and texture enhancement than traditional integer-based algorithms. Inspired by the [36], an R-L fractional differential operator is implemented to enhance the contrast and texture features for edge detection in order to provide precise visual features for the following path planning framework. To save computing resources and accelerate the edge feature extraction, we only employ the R-L fractional-order differentiation operation on a predefined region of interest (ROI), as shown in the rectangular area in the different camera views in Fig 2.

Formally, given an $M\times N$-resolution ROI image $\mathcal{R}_{M\times N}$, let p_x and p_y denote the pixel's location, d_x and d_y as the distance between the pixels in x and y direction, respectively. Then the relationship between the pixel and distance is formulated as $p_x = d_x i$ and $p_y = d_y j$. On the basis of the R-L definition, α-order fractional derivative approach is performed at each point of p_x and p_y. An α-order fractional derivative of $\mathcal{R}(i,j)$ with respect to p_x can be computed as:

$$\mathcal{R}^{\alpha}_{p_x} = \frac{\nabla^{\alpha}\mathcal{R}\left(p_x,p_y\right)}{d^{\alpha}_x} = d_x^{-\alpha}\sum_{i=1}^{M}\sum_{j=1}^{N} w^{\alpha}(i,j)\mathcal{R}(i,-(M+N)) \tag{1}$$

and a α-order fractional derivative of $\mathcal{R}(i,j)$ with respect to p_y can be represented as:

$$\mathcal{R}^{\alpha}_{p_y} = \frac{\nabla^{\alpha}\mathcal{R}\left(p_x,p_y\right)}{d^{\alpha}_y} = d_y^{-\alpha}\sum_{i=1}^{M}\sum_{j=1}^{N} w^{\alpha}(i,j)\mathcal{R}(-(M+N),j) \tag{2}$$

where $w^{\alpha}(i,j)$ is the coefficient at pixel space (i,j) and denoted as $w^{\alpha}(i,j) = (-1)^{i+j}\begin{pmatrix}\alpha\\ i+j\end{pmatrix}$. Then, we can denote the α-order derivative of $\mathcal{R}(i,j)$ with respect to p_x and p_y in a matrix form as:

$$\begin{aligned}&\begin{pmatrix}\left(d_x+d_y\right)^{-\alpha}\nabla^{\alpha}\mathcal{R}(0,0) & \dots & \left(d_x+d_y\right)^{-\alpha}\nabla^{\alpha}\mathcal{R}(0,N)\\ \vdots & \ddots & \vdots\\ \left(d_x+d_y\right)^{-\alpha}\nabla^{\alpha}\mathcal{R}(M,0) & \dots & \left(d_x+d_y\right)^{-\alpha}\nabla^{\alpha}\mathcal{R}(M,N)\end{pmatrix}\\ &= \frac{1}{\left(d_x+d_y\right)^{\alpha}}\begin{pmatrix}w^{\alpha}(0,0) & \dots & w^{\alpha}(0,N)\\ \vdots & \ddots & \vdots\\ w^{\alpha}(M,0) & \dots & w^{\alpha}(M,N)\end{pmatrix}\\ &\begin{pmatrix}\mathcal{R}(0,0) & \dots & \mathcal{R}(0,N)\\ \vdots & \ddots & \vdots\\ \mathcal{R}(M,0) & \dots & \mathcal{R}(M,N)\end{pmatrix}\end{aligned} \tag{3}$$

On the basis of the matrix form of the R-L α-order derivative with respect to p_x and p_y, we compute an edge vector $\boldsymbol{E}$ in Eq. (4) by convolutions of input ROI image with its α-order fractional derivative in x and y direction. As a result, we can denote each edge component as the convolution between the image and its corresponding fractional differential pair, formulated by $E_x(i,j) = \mathcal{R}^{\alpha}_{p_x}\otimes\mathcal{R}(i,j)$ and $E_y(i,j) = \mathcal{R}^{\alpha}_{p_y}\otimes\mathcal{R}(i,j)$, where $E_x(i,j)$ and $E_y(i,j)$ are the edge components in x and y directions, respectively, and $\otimes$ is the convolution operator.

$$\boldsymbol{E} = \frac{1}{\left(d_x+d_y\right)}\left[\left\{\mathcal{R}^{\alpha}_{p_x}\otimes\mathcal{R}(i,j)\right\}\boldsymbol{x} + \left\{\mathcal{R}^{\alpha}_{p_y}\otimes\mathcal{R}(i,j)\right\}\boldsymbol{y}\right] \tag{4}$$

$$d_x + d_y = \max_{i,j}\left(\sqrt{\left(\mathcal{R}^{\alpha}_{p_x}\right)^2 + \left(\mathcal{R}^{\alpha}_{p_y}\right)^2}\right) \tag{5}$$

As the magnitude and direction of the edge vector $\boldsymbol{E}$ may not be distributed uniformly in unclear ROI images, we average all values that are close to one another to improve edge identification. Therefore, the magnitudes of edge component $E(i,j)|$ and directional component $dir[E(i,j)]$ are represented as:

$$|E(i,j)| = \frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}\sqrt{(E_x(i,j))^2 + \left(E_y(i,j)\right)^2} \tag{6}$$

$$dir[E(i,j)] = \sum_{i=1}^{M}\sum_{j=1}^{N}\tan^{-1}\left(\frac{E_y(i,j)}{E_x(i,j)}\right) \tag{7}$$

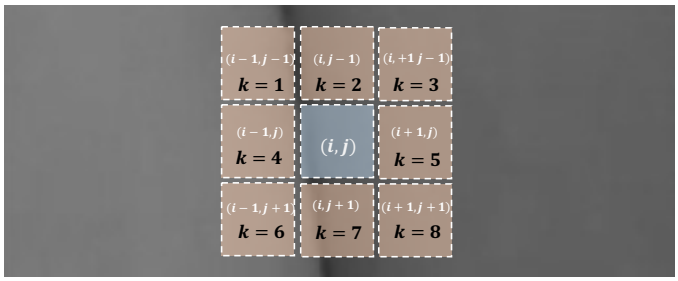

Fig. 4: Conceptual representation of edge computations for the edge intensity generation. The edge intensity with respect to a position (i, j) in the ROI image is considered to compute all the neighbors in a matrix with a (3×3) size from the perspectives of the edge magnitude, directional component, and contrast.

The edge vector is used in the proposed edge detection pipeline. Edge magnitude is used as a first measure, the directional component is used as a second measure, and contrast is used as a third measure in order to increase the probability of searching for the strong and proper border. As shown in Fig. 4, the succeeding position with respect to a position (i, j) of the ROI image is calculated in a matrix with a $(h \times w)$ size (normally, 3×3). With the aid of edge magnitude, directional component, and contrast, the edge intensity of its k-th surrounding neighbor $I_k(h, w)$, $k = 1, 2, \ldots 8$ can be computed using Eq. (8) from eight directions as:

$$I_k(h, w) = |E_{ij}(h, w)| + dir[E_{ij}(h, w)] + C_{ij}(h, w) \tag{8}$$

where $|E_{ij}(h, w)|$, $dir[E_{ij}(h, w)]$, $C_{ij}(h, w)$ denote the edge component, the directional component, and contrast component respectively, and each item can be calculated as follows:

$$|E_{ij}(h, w)| = \frac{|E(i + h - 1, j + w - 1)|}{\max_{i,j} |E(i, j)|} \tag{9}$$

$$dir[E_{ij}(h, w)] = 1 - \frac{dir[E(i, j)] - dir[E(i + h - 1, j + w - 1)]}{\pi} \tag{10}$$

$$\begin{aligned} C_{ij}(h, w) &= \Big(S_f(i, j) - S_b(i, j)\Big) \\ &\times \ln\left(\frac{\bar{g}_b(i + h - 1, j + w - 1) \bar{\ominus} \bar{g}_f(i + h - 1, j + w - 1)}{\bar{g}_b(i + h - 1, j + w - 1) \bar{\oplus} \bar{g}_f(i + h - 1, j + w - 1)}\right) \end{aligned} \tag{11}$$

For each edge intensity computation, the three components are computed to find relative values within the surrounding pixel space $(i+u-1, j+v-1)$, and a large value of $I_k(h, w)$ indicates a higher possibility of edge existence in the corresponding direction. In the contrast component calculation, the foreground and background image entropy are denoted as $S_f(i, j)$ and $S_b(i, j)$, Logarithmic Image Processing (LIP) subtraction and addition are denoted as $\bar{\ominus}$ and $\bar{\oplus}$, and the average gray values of foreground and background are denoted as $\bar{g}_f$ and $\bar{g}_b$, respectively. Based on the gray levels of the foreground and background ROI image, the contrast component is used to enhance the pixel contrast after computing the magnitude and direction of the strong edge.

For each pixel (i, j), an edge intensity $I_{ij}(h, w)$ is calculated on the basis of the mask matrix (h, w), and we will collect the pixel coordinates only when its edge intensity exceeds a predefined threshold τ. As shown in Fig. 3 (b), after performing on the entire ROI, a final candidate edge point set $\{(i, j)\}$ is formed to estimate the sealing seam e_1e_2 by using the RANSAC algorithm. With the resulting seam line equation, the two ending points, $(x_1, y_1), (x_2, y_2)$, intersecting with the rectangular boundary, can be easily computed.

Algorithm 1: Robust seam tracking with Kalman filter

Input: Observation of the ROI at time step t, $\mathcal{R}_t$
Output: ROI features at time step t , $d_{attr}(t)$ and $\theta(t)$.

for *each time step t* **do**
  // 1) Compute seam line points
  for *each pixel* (i, j) **do**
    Compute $I_{ij}(u, v)$ in the mask using Equ. (8);
    Collect candidate edge point (i, j) if $I_{ij}(u, v) \geq \tau$
  Compute the seam line equation with RANSAC algorithm on candidate edge point set $\{(i, j)\}$;
  Compute the two intersection points between the ROI rectangular and the seam line, $(x_1, y_1), (x_2, y_2)$;
  // 2) Prediction Time Update
  Predict state vector and error covariance:
  $\hat{X}_{t|t-1} \leftarrow F_t \hat{X}_{t-1|t-1} + W_t$
  $\hat{P}_{t|t-1} = F_t \hat{P}_{t-1|t-1} F_t^T + Q_t$
  // 3) Observation and Update
  Compute Kalman gain:
  $K_t = \hat{P}_{t|t-1} H_t^T \left(H_t \hat{P}_{t|t-1} H_t^T + R_t\right)^{-1}$
  Correction based on observation:
  $z_t = H_t X_t + v_t$
  $\widehat{X}_{t|t} = \hat{X}_{t|t-1} + K_t \left(z_t - H_t \widehat{X}_{t|t-1}\right)$
  $\hat{P}_{t|t} = (I - K_t H_t) \widehat{P}_{t|t-1}$
  Compute ROI features $d_{attr}(t)$ and $\theta(t)$ based on $\widehat{X}_{t|t}$;
  return $d_{attr}(t)$ and $\theta(t)$;

2.1.1. Kalman Filter

As illustrated in Fig. 3, we assume that the ROI must contain the sealing seam. Hence, seam tracking under two different camera perspectives is established. To further reduce the seam line tracking error, two Kalman filters are performed to predict and automatically correct sealing seam states for different camera views. To describe the sealing seam on a two-dimensional ROI image, we define the coordinate of two ending points as the parameters of the sealing seam. Therefore, the position and velocity of a sealing seam in Kalman filter can be denoted as (x_1, y_1, x_2, y_2), and $\left(v_{x_1}, v_{y_1}, v_{x_2}, v_{y_2}\right)$ respectively, the state vector is then expressed $X_t = (x_1^t, y_1^t, x_2^t, y_2^t, v_{x_1}^t, v_{y_1}^t, v_{x_2}^t, v_{y_2}^t)^T$, where t indicates the discrete time step over ROI frame intervals. Since the measurement consists of positions of two the seam ending points, the measurement vector can be expressed $z_t = (x_1^t, y_1^t, x_2^t, y_2^t)^T$, and the transition matrix F_t and measurement matrix H_t in this seam tracking system can be denoted as:

$$F_t = \begin{pmatrix} \mathbf{I}_4 & \Delta t \mathbf{I}_4 \\ \mathbf{0} & \mathbf{I}_4 \end{pmatrix} \quad \text{and} \quad H_t = \begin{pmatrix} \mathbf{I}_4, \mathbf{0} \end{pmatrix} \tag{12}$$

where $\mathbf{I}_4 = \text{diag}(1, 1, 1, 1)$.

The Kalman filter predicts the state by using the model system equations, which are based on the interaction between the sealing gun tip and the sealing workpiece, and the previous state matrix. Then we can predict the parameters of the next state seam. When updating the time step, the Kalman filter leverages the measurement generated from our designed edge detector to refresh the predicted state for determining the position of the sealing seam in ROIs. The major goal of this tracking system is to estimate the optimal X_t based on the measurement z_t. Considering the above description, our robust seam tracking with Kalman filter from one camera perspective is illustrated in Alg. 1. After the measurement of the sealing seam, it involves two main stages, which are prediction time update and observation and update processes.

1) Prediction time update: After initializing the parameters of the related state vector, the prediction equations are defined as the Phrase (2) in the algorithm, where the transition matrix is defined as F_t, and we perform this effect for each state vector. The zero mean white prediction noise with covariance matrix Q_t is denoted as W_t. Last, the noise covariance between different terms in the state vector is represented as $\hat{P}_t$.

2) Observation and Update: In this phase, the algorithm deals with the computation of the Kalman gain K_t, the ratio between the covariance of predicted noise and the covariance of measured noise. Subsequently, we update the prediction model. During the updating process, the measurement matrix is represented as H_t represents, and v_t is a zero mean white measured noise with covariance R_t. At last, corrected seam line points can be obtained to compute the ROI features, $d_{attr}(t)$ and $\theta(t)$, for the motion policy generation, which we will illustrate in the following section.

2.2. RMP for Autonomous Sealing

In this part, we begin with a brief overview of RMP and its application to robotic sealing control (see [35] for a theoretical introduction), then followed by the RMP controller design for our robotic sealing scenario and our proposed neural RMPs.

2.2.1. RMP Modeling

Consider a robotic agent $\mathbf{x}$ in the task space $\mathbb{R}^n$ (usually $n = 3$ for the robotic arm system), the position of the end-effector of the robot arm at time step t is defined as $\boldsymbol{x}(t)$. In RMP, the motion policy is defined as a mapping from the position and velocity of the robot to acceleration, which is denoted by $f_{motion} : \boldsymbol{x}(t), \dot{\boldsymbol{x}}(t) \rightarrow \ddot{\boldsymbol{x}}(t)$. The robot moves to a new state $\boldsymbol{x}(t+1), \dot{\boldsymbol{x}}(t+1)$ by using the acceleration policy for a short time period, from which a trajectory can be formed by using forward integration. In autonomous sealing tasks, we attempt to establish a motion policy to move the robot to reach a goal configuration $g \in \mathbb{R}^n$ while moving along the sealing seam and avoiding collisions in the sealing grooves. To this end, we model desired behaviors of the sealing gun tip as reaching the nearest sealing seam and adjusting the orientation of the heading direction. As shown in Fig. 3, the nearest reaching point is the intersection point p_a between the sealing seam, e_1e_2, and p_tp_a. On the other hand, the seam angle is computed between the sealing seam, e_1e_2, and the vertical line or horizontal line. And these ROI features can be directly measurable using the proposed Alg. 1. Then the problem becomes designing a motion policy that controls the heading direction to ensure a small seam angle θ while attracting the sealing gun tip to the generated attraction point p_{attr} on the basis of the seam line.

Intuitively, the motion policy can be easily designed by dividing it into a collection of different policies that model the interaction between the sealing gun tip and attraction point or sealing angle from different perspectives. For example, for each attraction point, we design a policy that generates an attractive acceleration towards the computed attraction point, while for the sealing angle, the policy attempts to form a rotation acceleration to control the heading direction. Finally, by solving a least-squares problem, the resulting motion policy can be computed analytically as below:

$$\boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \underset{\ddot{\boldsymbol{x}}}{\text{argmin}} \sum_i \frac{1}{2} \|\boldsymbol{f}_i(\boldsymbol{x}, \dot{\boldsymbol{x}}) - \ddot{\boldsymbol{x}}\|^2 \tag{13}$$

where $\boldsymbol{f}_i$ represents the acceleration policy of the i-th attraction behavior or sealing angle control behavior. However, Eq. 13 may lead to negative behaviors. For instance, it ignores the error heading direction. When the robot goes parallel to the sealing groove as opposed to heading towards the sealing seam, this can make a significant impact. A sound solution is to employ a Riemannian metric to expand the local space so that the cost of the behavior of going in one direction differs from that of moving in another. By introducing a Riemannian metric $\boldsymbol{A}$ to depict the magnitude of a cost vector $\boldsymbol{v}$, denoted by $\|\boldsymbol{v}\|_{\boldsymbol{A}}^2 = \boldsymbol{v}^\top \boldsymbol{A} \boldsymbol{v}$, the policy is reformulated as:

$$\boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \underset{\ddot{\boldsymbol{x}}}{\text{argmin}} \sum_i \frac{1}{2} \|\boldsymbol{f}_i - \ddot{\boldsymbol{x}}\|_{\boldsymbol{A}_i}^2 \tag{14}$$

where $\boldsymbol{A}_i$ represents the Riemannian metric with respect to its corresponding policy $\boldsymbol{f}_i$. By doing so, we expand the original motion policy into a Riemannian Motion Policy, and for each policy $\boldsymbol{f}_i$, it has an associated Riemannian metric $\boldsymbol{A}_i$.

Although Eq. (14) has considered the end-effector of the robot, the physical robotic sealing system has non-negligible shapes (e.g., a sealing gun) and complex mechanics (e.g., a multiple DOF robotic arm). Taking these extra considerations into account, we define the robot as a collection of control points and sealing seam angles, $\boldsymbol{x}_1^{tip}, \boldsymbol{x}_2^{tip}, \cdots$, with corresponding forward kinematic functions (also known as a task map) ϕ linked to a joint configuration space $\boldsymbol{q}$ defined by $\boldsymbol{x} = \phi(\boldsymbol{q})$. Considering a classic 6-DoF robot consisting of six joints, denoted by $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$, the transformation matrix from the base to the robot hand, ${}^bT_h$, and also known as its forward dynamics model, can be denoted as:

$${}^bT_h = {}^0\phi_6(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

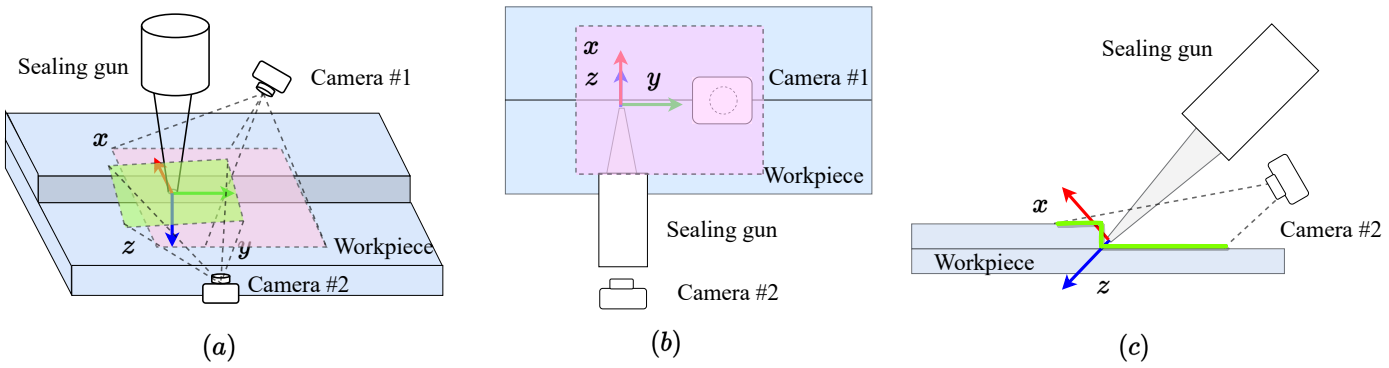

Fig. 5: Conceptual representation of the robotic sealing system with ROIs from two different camera views. (a) visualizes an overview of the robotic sealing configurations from a third-person perspective, (b) shows a top-down view of the robotic sealing configurations, and (c) presents a side view along the sealing heading direction.

While the transformation from the sealing gun tip to the robot base frame, ${}^{b}\mathbf{T}_{p}$, is given by:

$$ {}^{b}\mathbf{T}_{tip} = {}^{b}\mathbf{T}_{h} \cdot {}^{h}\mathbf{T}_{tip} \tag{16} $$

where ${}^{h}\mathbf{T}_{tip}$ denotes the transformation from the sealing gun tip to the robot hand. Since the sealing gun tip is defined as our Tool Center Point (TCP), we keep its orientation as same as the robot hand. Therefore, the final transformation of the sealing gun tip with respect to the robot base can be regarded as a translation $\mathrm{Trans}(d_x, d_y, d_z)$ (d_i denoting the fixed translation distance along the i-th axis) from the robot hand to the sealing tip, denoted by:

$$ \mathrm{Trans}(d_x, d_y, d_z) \times {}^{b}T_{h} = \begin{pmatrix} n_x & o_x & a_x & p_x + d_x \\ n_y & o_y & a_y & p_y + d_y \\ n_z & o_z & a_z & p_z + d_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{17} $$

where

$$ \mathrm{Trans}(d_x, d_y, d_z) = \begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{18} $$

Therefore, the final forward kinematic function for our robotic sealing task can be represented as:

$$ \boldsymbol{x} = \phi(\boldsymbol{q}) = \phi(\theta_1, \cdots, \theta_6, d_x, d_y, d_z) \tag{19} $$

By combining the Jacobian of the kinematic function $\boldsymbol{J}_{\phi} = \frac{\partial \phi}{\partial \boldsymbol{q}}$ into Eq. (14), we can compute the optimal RMP policy in the joint space as below:

$$ \boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \underset{\ddot{\boldsymbol{q}}}{\operatorname{argmin}} \sum_{i} \frac{1}{2} \| \boldsymbol{f}_i - \boldsymbol{J}_{\phi_i} \ddot{\boldsymbol{q}} \|^2_{\boldsymbol{A}_i} \tag{20} $$

By introducing the Jacobian of a robot control point involved in motion policy i, which is defined as $\boldsymbol{J}_{\phi_i}$, we can solve the Eq. (20) analytically as below:

$$ \boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \ddot{\boldsymbol{q}}^{*} = \left(\sum_{i} \boldsymbol{J}_i^{\top} \boldsymbol{A}_i \boldsymbol{J}_i \right)^{+} \left(\sum_{i} \boldsymbol{J}_i^{\top} \boldsymbol{A}_i \boldsymbol{f}_i \right) \tag{21} $$

where + denotes pseudo-inverse.

2.2.2. *Locally Reactive RMP Controller Design*

The overview configuration of the proposed robotic sealing application is illustrated in Fig. 5, using two monocular cameras observing the forward and side view of the interaction between the sealing gun tip and the local environment. Since our approach is aimed at solving the entire sealing path planning task by repeatedly generating new local desired robot motions, so essentially, we are interested in the locally reactive RMPs. Tab. 1 summarizes all the RMPs used for the design of the reactive controller and the corresponding formulations for the fields and metrics. However, before implementing and integrating them with the robotic sealing system, there are still factors to consider:

1) Reference frames and state representation: Fig. 5(a) illustrates the frame conventions considered in our formulation. The state $\boldsymbol{x}$ of the robotic sealing device is given by the pose of the sealing gun tip $\boldsymbol{T}_{tip} \in \mathrm{SE}(3)$ expressed in the fixed TCP frame F_{tip}, and the state of the ROI features d_{attr} and θ are expressed in the image frame F_{img}. The velocity $\dot{\boldsymbol{x}}$ and acceleration $\ddot{\boldsymbol{x}}$ of the sealing gun tip are represented in the TCP frame as $\boldsymbol{v}_{tip} = (v_x, v_y, v_z, v_\alpha, v_\beta, v_\gamma) \in \mathbb{R}^3$ and $\boldsymbol{a}_{tip} = (a_x, a_y, a_z, a_\alpha, a_\beta, a_\gamma) \in \mathbb{R}^3$, respectively.

2) RMPs in different views: Our autonomous sealing employs two different views to generate multiple RMPs so that it can recover the motion generation in the original 3D working space. For each view, two specific RMPs are designed, where one is attraction RMP, and the other is heading correction RMP. The attraction RMP aims to minimize the position between the sealing gun tip and the attraction point in the ROI, and it only affects the translation part of the motion policy. While the heading correction RMP attempts to move to the goal point and minimize the sealing seam angle in the ROI, which only works with the orientation part. As illustrated in Tab. 1, we use $\rightarrow$ and $\downarrow$ to represent the RMPs in forward and side views, respectively. For *Attraction RMPs*, the transformation from the image space to the TCP frame ${}^{\mathrm{F}_{img}}\boldsymbol{T}_{\mathrm{F}_{tip}}$ (for simplicity, we use ${}^{img}\boldsymbol{T}_{tip}$ in the following) varies and depends on the different views. As shown in Fig. 6d, the d_{attr} can be divided into d^{x}_{attr} and d^{y}_{attr} by projecting along the x and y axis in F_{img}. In the forward view, d^{x}_{attr} is only considered to have an impact on the x-directional translation for the sealing gun. On the other hand, d^{y}_{attr} is used for side-view

Table 1: Main Riemannian Motion Policies (RMP) used for the reactive controller formulation

RMP	Attraction (→) †	Heading Correction (→) †	Attraction (↓)†	Heading Correction (↓) †	Damping	Regularization
Descriptrion	Minimizes the position between the sealing gun tip and the attraction point in forward view	Moves to the goal point and minimizes the ROI angle in forward view	Minimizes the position between the sealing gun tip and the attraction point in side view	Moves to the goal point and minimizes the ROI angle in side view	Damps the output acceleration depending on the current velocity	Smooths the output acceleration using the last acceleration command
Applied to	TCP	TCP	TCP	TCP	TCP	TCP
Acceleration field f	$-k\nabla f_{attr}({}^{tcp}\mathrm{T}_{img})d_{attr}$ ‡	$kf_{rot}({}^{tcp}\mathrm{T}_{img})\theta$ ‡	$-k\nabla f_{attr}({}^{tcp}\mathrm{T}_{img})d_{attr}$ ‡	$kf_{rot}({}^{tcp}\mathrm{T}_{img})\theta$ ‡	$-k\boldsymbol{V}_{tip}$	$\boldsymbol{a}_{tip}(t-1)$
Components affected	Only translation, ${}_{tip}f_{\alpha,\beta,\gamma}=0$	Only orientation, ${}_{tip}f_{x,y,z}=0$	Only translation, ${}_{tip}f_{\alpha,\beta,\gamma}=0$	Only orientation, ${}_{tip}f_{x,y,z}=0$	Translation and orientation	Translation and orientation
Metric M	$\sigma(d,d_c)\,\mathbf{I}_{3\times3}$ *	$\sigma(\theta,\theta_c)\,\mathbf{I}_{3\times3}$ *	$\sigma(d,d_c)\,\mathbf{I}_{3\times3}$ *	$\sigma(\theta,\theta_c)\,\mathbf{I}_{3\times3}$ *	$\sigma(d,d_c)\,\mathbf{I}_{6\times6}$ *	$s\mathbf{I}_{6\times6}$
When Enabled?	Distance to attraction point, d_{attr}, is further than d_c	Vertical angle in the ROI, θ, is further than θ_c	Distance to attraction point, d_{attr}, is further than d_c	Horizontal angle in the ROI, θ, is further than θ_c	Always	Always

† → and ↓ represent the forward view and side view for the robotic sealing, respectively.
‡ ${}^{tcp}\mathrm{T}_{img}$ denotes the transformation from the image frame to the tool center point frame.
* The function $\sigma(d)$ is implemented as a logistic/inverse logistic function that becomes 1 when d is closer/farther than d_c depending on the case.

RMPs. For *Heading RMPs*, θ is only considered to deal with the rotation acceleration along the z-axis a_γ in frame F_{tip} in the forward view, and a_α for side view.

By combining the RMPs summarized in Tab. 1, we can compute our final optimal acceleration of the sealing gun tip in F_{tip} frame with the below equation:

$$\arg\min_{\mathrm{F}_{tip}} \underbrace{\sum_{\mathrm{F}_{tip},\boldsymbol{M}} \left\| {}^{\mathrm{F}_{tip}}\mathbf{f} - {}^{\mathrm{F}_{tip}}\mathbf{a} \right\|^2_{\mathbf{M}}}_{\text{RMPs employed in the TCP}} \tag{22}$$

where we expand the original RMP formulation in Eq. (14) to a motion policy where each case is explicitly incorporated into.

2.3. Neural RMPs

In this section, we propose two neural models to learn robot controllers from visual ROI images for robotic sealing tasks. One model is to predict the robot control command directly with the neural net, which is also selected as a baseline model to compare our model that instead predicts the parameter for generating RMPs first and then performing sealing gun controls.

Neural controls: An intuitive strategy is to train an end-to-end neural network that directly predicts the control commands used for sealing gun behavior. This is intriguing because it can learn geometric and semantic information from visual representations. However, this technique is totally data-driven. Thus it lacks an understanding of how the environment influences the sealing gun's behavior. Furthermore, without explicitly modeling the geometry and dynamics of the robotic sealing system, it could also have a negative effect on the generalizability of the model application, which we will illustrate in the following experiments.

Neural RMPs: To overcome the shortcomings of the preceding method, we offer a novel model that predicts RMPs from seam features extracted in ROI images. In the given design, the

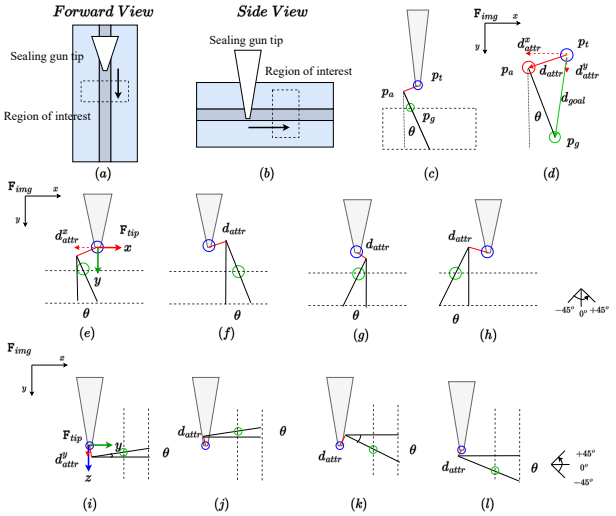

Fig. 6: Conceptual representation of different state representations for RMPs modeling process. (a) and (b) show the forward and side views of the sealing process. (c) and (d) depicts an example of an enlarged interaction pattern in the forward view. (e)–(h) present the different interactions in the forward view. (i)–(l) display the different interactions in the side view.

command applied to the control point of the sealing gun tip is an RMP acceleration component $\boldsymbol{f}$ and its associated Riemannian metric $\mathbf{A}$ that encodes the locally joint geometry of the sealing gun and the environment. Furthermore, by combining the Jacobian matrix and the kinematic model of the predefined control point, the final optimal control command for the sealing gun's behavior can be solved by combining the RMP for each control point with a unified geometrical and kinematical representation, thus resulting in a more generalizable and interpretable model, especially on unseen dynamic environments.

Fig. 7 depicts the conceptual representation of the framework of the propose neural RMPs and the neural control baseline model. Different from the neural control model, apart from multiple feature extractors and a regressor, our neural RMPs have an additional solver to compute the optimal acceleration commands. The image feature extractor is completed with our designed seam detection pipeline. Since an RMP may need extra input data, such as the velocity and angular velocity of the sealing gun tip, we employ multiple fully connected layers to extract the features for each extra input. Following that, to predict RMPs parameters (i.e., accelerations and Riemannian metrics), the extracted features are concatenated and input into a regressor, which is composed of multiple fully connected layers. According to the definitions of RMP [35], several RMPs can be defined on one sealing gun control point and combined into one single equivalent RMP based on the addition operator designed under the RMP architecture. Mathematically, let two RMPs in space Φ be $\mathcal{M}_1 = {}^{\Phi}(\boldsymbol{f}_1, \boldsymbol{A}_1)$ and $\mathcal{M}_2 = {}^{\Phi}(\boldsymbol{f}_2, \boldsymbol{A}_2)$, then we can combine them as follows:

$$\mathcal{M}_1 + \mathcal{M}_2 = \left((\boldsymbol{A}_1 + \boldsymbol{A}_2)^{+} (\boldsymbol{A}_1\boldsymbol{f}_1 + \boldsymbol{A}_2\boldsymbol{f}_2), \boldsymbol{A}_1 + \boldsymbol{A}_2\right)_{\Phi} \quad (23)$$

where + denotes the pseudo-inverse and will reduce to the inverse with full rank. In general, we can combine multiple RMPs $\{\mathcal{M}_i\}_{i=1}^{n}$ into a single RMP $\mathcal{M}_* = {}^{\Phi}(\boldsymbol{f}_*, \boldsymbol{A}_*)$ by averaging the weighted Riemannian metric:

$$\mathcal{M}_* = \sum_i \mathcal{M}_i = \Phi\left(\left(\sum_i \boldsymbol{A}_i\right)^{+} \sum_i \boldsymbol{A}_i\boldsymbol{f}_i, \sum_i \boldsymbol{A}_i\right) \quad (24)$$

which generates the optimal acceleration policy for the combined autonomous sealing system. Specifically, in our designed RMPs, since each metric is of the form $\boldsymbol{A}_i = w_i\boldsymbol{I}$, then it reduces to the classic weighted average specified by: $(\boldsymbol{f}_*, \boldsymbol{A}_*) = \left(\frac{1}{w}\sum_i w_i\boldsymbol{f}_i, \frac{1}{w}\sum_i w_i\boldsymbol{A}_i\right)$, where $w \equiv \sum_i w_i$. As a result, only one RMP needs to be predicted for each control point of the sealing gun.

In the robotic sealing system, an RMP is composed of a 6-element acceleration $\mathbf{f}$ and the associated 6×6 Riemannian metric matrix. We only predict half element values of the Riemannian metric due to its symmetric property and then construct the entire matrix. Note that since some of our designed RMP used for locally reactive control are designed to only have an effect on translation or orientation components, the predicted acceleration and Riemannian metric will be reduced in a half accordingly. Last but not least, the solver in our model is not a component of the network, which is only used to compute the final optimal robot control command for the sealing system.

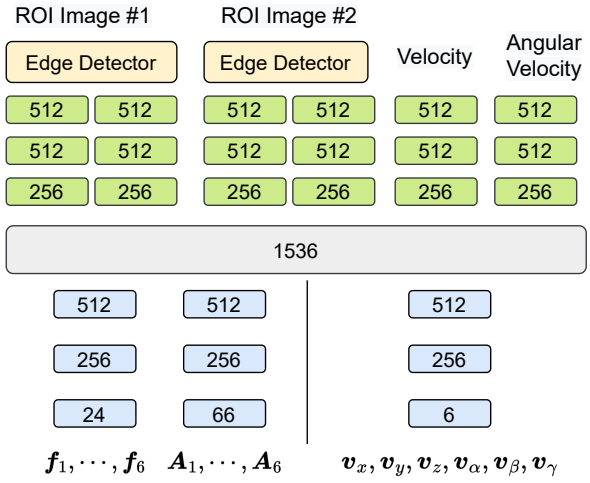

Fig. 7: Our RMP network and a baseline model. The predicting controls network reuses the seam feature extractor of the RMP network.

To compare with our proposed neural RMPs model, a baseline model that predicts sealing gun controls is also implemented (see Fig. 7 right part), where we employ a regressor to directly output the control command for the sealing gun instead of RMP parameters. To examine which approach is more effective and generalizable, we maintain the architecture and computing cost for both models at the same level.

3. System Implementation

To validate the performance of the proposed reactive sealing path planning method, we built a robotic sealing system based on a UR-5 robot manipulator and a modified pneumatic sealing pipeline. In the following, we will discuss the system construction in detail and followed by the ROS-based software platform.

3.1. System construction

To automate the robotic sealing process, a pneumatic system is built as illustrated in Fig. 9. Our air supply source for the pneumatic system is a Jun-air 12-40 compressor, which is suitable for common laboratory applications and gas generation despite having a 40-Litre receiver size and 1200W compressor motors. Manual control valves are used to measure the real-time air pressure in the mainstream and are also connected to the control box of a UR-5 robot so that we can control the airflow by assigning the control input signal in python code. The sealing gun is the pneumatic actuator that converts air pressure into physical sealant pushing motion. We modify the original sealing gun by removing the gunstock grip and connecting the air intake with the air compressor. Instead, we 3D-print a supporting holder to install the rest sealing gun and mount it on the end-effector of the UR robot. Also, two monocular cameras are mounted on the front and side view of the sealing gun tip. The final experiment setup is shown in Fig. 9.

Generally, in order to choose the correct silicone sealant, one needs to take into account the type of cure and more importantly the modulus. Though there are many main types of silicone sealant, such as High Modulus Acetoxy, Low Modulus

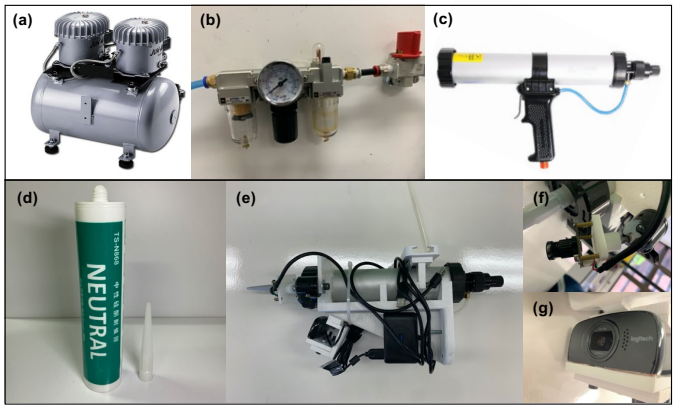

Fig. 8: The designed pneumatic system for robotic sealing tasks is composed of (a) Jun-air 12-40 air compressor, (b) manual control Valve, (c) original pneumatic sealing gun, (d) silicone sealant, (e) modified pneumatic sealing gun, and (d) two camera sensors.

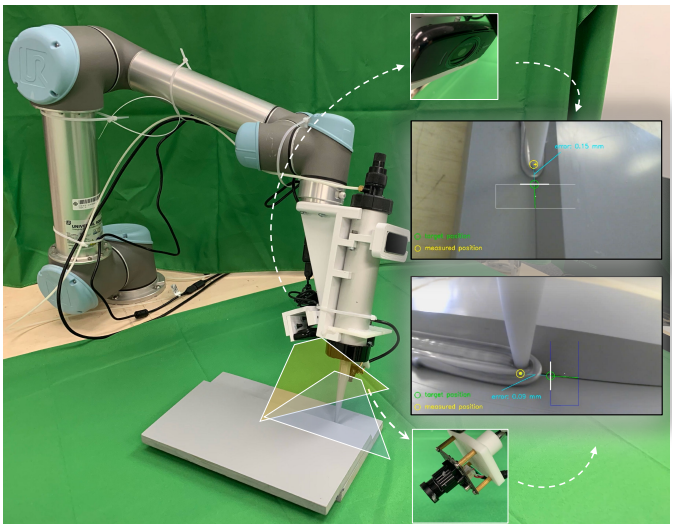

Fig. 9: The experimental set-up for two-camera-based robotic sealing tasks.

Neutral, Low Modulus Acetoxy, and so on, the neutral cure silicone sealant, as shown in Fig. 8d, is used for our experiments. Because this type of silicone sealant deposits methyl ethyl ketoxime during curing, which is a non-corrosive, thixotropic substance, thus making neutral cure silicones suited for electronics applications. Furthermore, although having a longer cure period than acetoxy cure silicones, these silicones have a considerably more faint odor, making them ideal for indoor applications such as kitchen installations. But considering our case, we need to clear out the sealant to do the experiment repetitively.

3.2. Software platform

To coordinate different sensor and signal inputs for the above-designed robotic system in a seamless manner, a software platform is mainly built based on the robot operating system (ROS). As shown in Fig. 10, preliminary settings (e.g., ROI corner and sealing tip positions in different views, IP address for connecting UR robot, TCP transformation matrix) are loaded from `sealing.conf` in front and side view

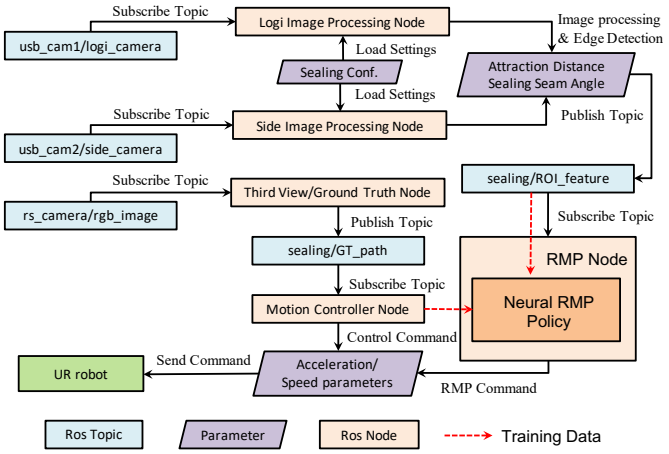

Fig. 10: The ROS architecture for the proposed two-camera-based neural path planning system for robotic sealing tasks.

image processing ROS nodes, which subscribe the ROS topics from the cameras providing front and side views, named `usb_cam1/logi_camera` and `usb_cam2/side_camera`, respectively. Based on the RGB ROI images, the proposed robust edge detection algorithm is implemented to compute the ROI features, including attraction distances and sealing seam angles in different views. Those features will be used to train the neural RMPs or execute the generated RMPs as their inputs. During the neural RMP training process, a top-down camera will be used to detect the AR code labeled on different workpieces by subscribing to the `rs_camera/rgb_image` topic. Since the relative transformation between the workpiece and the AR code position is calibrated with the geometrical calculation. By doing so, the ground truth sealing trajectory can be generated according to the sealing seam. The motion controller node deals with the generated ground truth sealing path and outputs the robot control command to the RMP node as the corresponding ground truth of the ROI features. Once the training process is completed, the RMP node will combine all the RMPs into one to send the resulting control commands to the UR robot. Note that the control commands not only contain the robot motions but also the air pressure controls.

4. Experiments

In this section, to verify the effectiveness of our proposed methods, three workpieces are considered to test with the designed robotic sealing system and the proposed neural RMP-based path planning method. As shown in Fig. 11 (a)-(c), the testing workpieces have different joint types, which are straight-line, cylinder, and mixed joints, respectively. The results of the ROI edge features performed with our designed edge detection pipeline are presented first, then with trained neural RMPs sealing performance on different workpieces is analyzed. More importantly, we conduct reactive path planning experiments under environmental disturbances to examine the local reactive behaviors. Last, the limitations of the proposed approach are also mentioned.

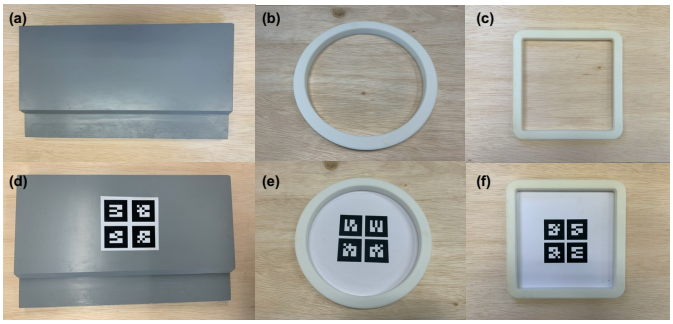

Fig. 11: (a)-(c) are flat workpiece, cylinder workpiece, and rectangular workpiece; (d)-(f) are corresponding workpieces labeled with AR codes.

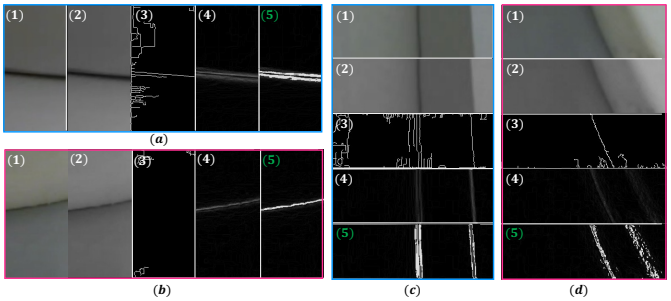

Fig. 12: (a) and (c): edge detection results of the flat workpiece in the ROIs of forward and side views. (b) and (d): edge detection results of the cylinder workpiece in the ROIs of forward and side views. For each subfigure, (1)-(5) denote the original ROI image, gray ROI image, Canny result, Sobel result, and our result.

4.1. Edge Detection Pipeline

Based on the designed edge detection pipeline, the R-L fractional-order differentiation edge detector is employed first to detect the sealing seam for the tested workpieces from the ROIs in two different views (forward and side views). The ROIs are in 100×50-pixel or 50×100-pixel size, and their positions in different views are loaded according to the parameter settings in the `sealing.conf`. All the experiments are performed with OpenCV python library on a PC with AMD R7/3.2GHZ/16GB specification. As shown in Fig. 12, two groups of results are randomly selected from the experiments, one is ROI images for flat workpiece (see Fig. 12 (a) and (c)) and the other is cylinder workpiece (Fig. 12 (b) and (d)). In each subfigure, (1) shows the original ROI images, (2) shows a transformed grayscale map with 256 gray levels for (1), (3)-(5) shows the texture results generated with two classic edge detection algorithms, Canny [37] and Sobel [38], and our R-L fractional-order edge detection algorithm, respectively. By comparing the different texture results, the R-L fractional-order edge detection algorithm performs best in all four different ROI images. On the contrary, the Canny detector tends to produce lots of noise edge points without a good fine-tuning of the threshold values. For the Sobel detector, though it can get rid of most of the noise points, the edge points are not as clear as the R-L fractional-order edge detector in low-SNR conditions.

After performing R-L fractional-order edge detection algorithm, we leverage on Kalman filter to obtain an accurate estimation of the ROI features. With detected edges in ROI, two endpoints could be determined to compute our designed

State	Flat		Cylinder		Rectangular	
	w/o.	w/.	w/o.	w/.	w/o.	w/.
$d_{attr}^{(\rightarrow)}$	2.52	**0.78**	3.08	**0.88**	3.69	**0.94**
$d_{attr}^{(\downarrow)}$	2.66	**0.66**	3.14	**0.85**	3.93	**0.99**
$\theta^{(\rightarrow)}$	1.63	**0.56**	2.59	**0.73**	2.92	**0.97**
$\theta^{(\downarrow)}$	1.57	**0.61**	2.41	**0.81**	3.05	**1.05**

Table 2: Kalman performance of the ROI features, $[d_{attr}^{(\rightarrow)}, \theta^{(\rightarrow)}, d_{attr}^{(\downarrow)}, \theta^{(\downarrow)}]$, on different workpieces in terms of RMSE normalized to each state.

ROI features, which are attraction distance, d_{attr} and sealing seam angle, θ. Since the ROIs are generated from two different perspective views (forward and side views), four states, $[d_{attr}^{(\rightarrow)}, \theta^{(\rightarrow)}, d_{attr}^{(\downarrow)}, \theta^{(\downarrow)}]$, are considered to be estimated by Kalman filter as the input in the following RMPs. To distinguish the performance of the edge detection with and without Kalman filter, we set a ground truth trajectory to collect ROI images on flat and cylinder workpieces for 10 seconds. During the process, the ROI features with Kalman filter are also performed. We keep each camera capturing the images at 30 FPS. Hence, each workpiece will generate 600 ROI images, and the entire process ends up with 1200 ROI images in total. We manually label two endpoints to denote the edges in the ROI images to compute the ground truth estimation of the ROI features, $[\hat{d}_{attr}^{(\rightarrow)}, \hat{\theta}^{(\rightarrow)}, \hat{d}_{attr}^{(\downarrow)}, \hat{\theta}^{(\downarrow)}]$. The performance of the state estimation is measured by computing the Root Mean Square Error (RMSE) between the estimation and ground truth, and table 4.1 summarizes the final results. The performance of the state estimation after using Kalman filter is clearly improved from the original range $[2.52, 3.93]$ to reduced range $[0.66, 0.99]$ in terms of the attraction distance measurement, $d_{attr}^{(\rightarrow)}$ and $d_{attr}^{(\downarrow)}$. For the sealing seam angle, the Kalman filter also reduces the RMSE from the original range $[1.57, 3.05]$ to a shrank range of $[0.56, 1.05]$. According to the principle of sealing for most applications [1], an error within 1 sealing angle will not affect the welding quality.

4.2. Neural RMP

We trained our neural RMP model in the real workpiece sealing environment. The ground truth trajectories are generated by calibrating the transformation between the position of the AR code labeled on the workpiece and the CAD geometrical model. As shown in Fig. 11, we fix the starting and ending point for the sealing trajectory due to the limited working space of the UR robot manipulator. Our training dataset consists of 2.4k trajectories generated from flat and cylinder workpieces, which includes the most sealing cases, i.e., line and curve cases. During training, we keep the consistent velocity for the sealing forward direction (y-direction), and randomize the offset velocity (x-direction) $\sim \mathcal{U}(-0.01, 0.01)$), approaching velocity (z-direction) $\sim \mathcal{U}(-0.01, 0.01)$), forward rotation velocity (z-direction) $\sim \mathcal{U}(-10^\circ, 10^\circ)$), offset rotation velocity (x-direction) $\sim \mathcal{U}(-10^\circ, 10^\circ)$), lighting conditions (contrast $\sim \mathcal{U}(-0.3, 0.3)$ and brightness $\sim \mathcal{U}(-0.3, 0.3)$). The L2 loss is used to compute the gradients. We train each model with the same training dataset and the same number of epochs and test them in position-changed flat and cylinder workpieces and an unseen rectangular workpiece.

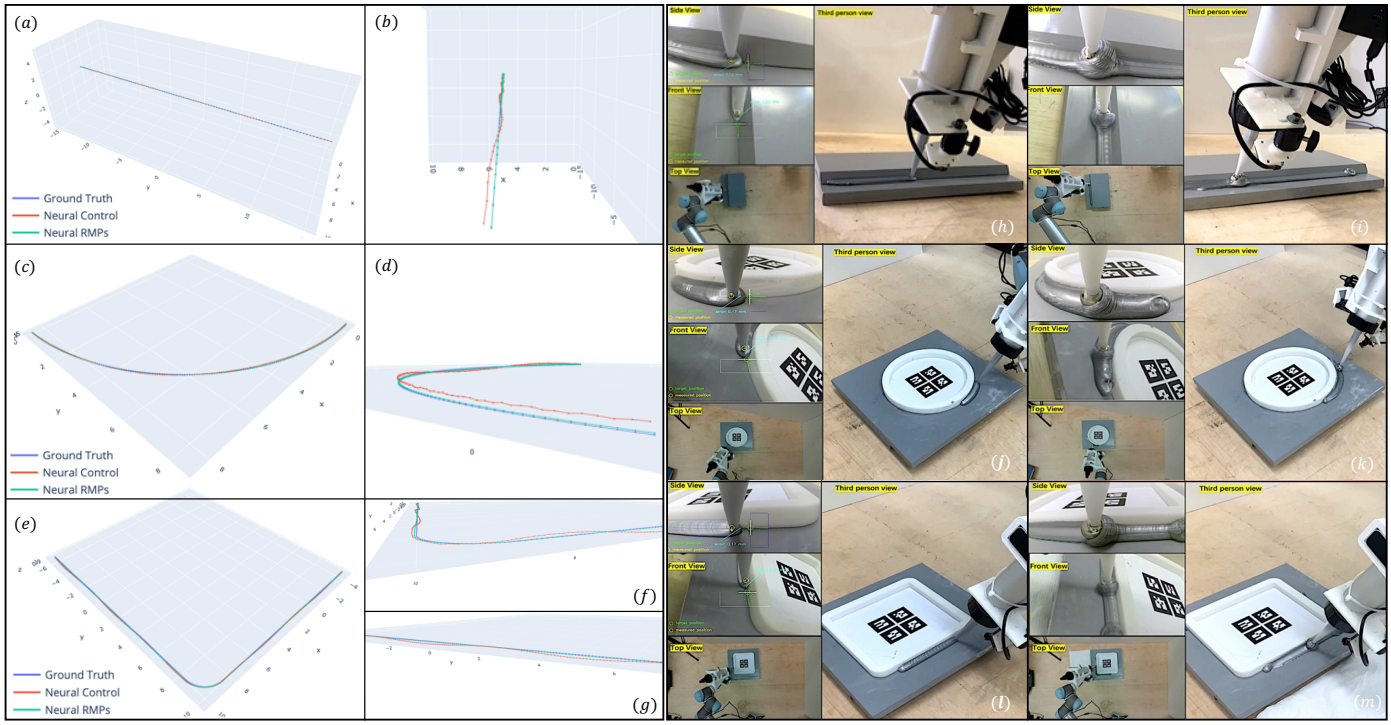

Fig. 13: (a), (c), and (e) are the sampled trajectories on three different sealing workpieces by different models, respectively, and (b), (d), (f), and (g) depict the detailed trajectory in a close side view. The ground truth is computed by calibrating the transformation between the AR code's position and the corresponding workpiece, which is marked in blue. The *Neural Controls* and *Neural RMPs* are marked in red and green, respectively. (h)-(i), (j)-(k), and (l)-(m) show the final sealing and smoothing experiments with our learned *Neural RMPs* model working on different sealing workpieces. For each case, two camera views (front and side views) are used to extract ROI features, and a top-down view is presented. The trajectories used for smoothing the silicone sealant are generated based on the previous sealing trajectories with an appropriate computed transformation matrix.

Methods	Flat			Cylinder			Rectangular		
	Collision	Reached	RMSE	Collision	Reached	RMSE	Collision	Reached	RMSE
Predicted Controls	8%	92%	2.36	6%	94%	2.17	40%	60%	4.43
Predicted RMPs	3%	97%	**0.51**	4%	96%	**0.63**	5%	**95%**	**0.71**

Table 3: Statistics on the three sealing workpiece environments. Note that the flat and cylinder workpiece is the training environment for our *Neural RMPs* and *Predicting Controls*, while the rectangular workpiece is an unseen environment to test their generalizability.

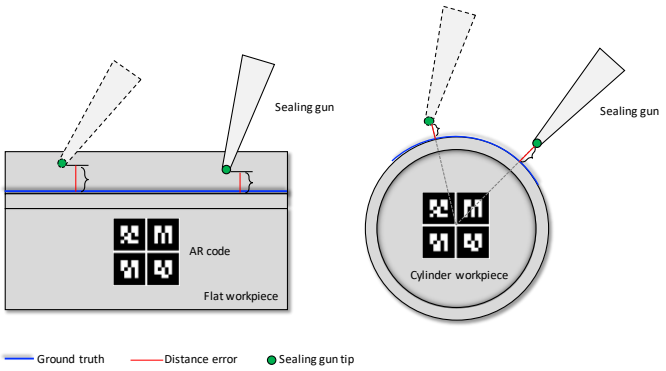

Fig. 14: Conceptual representation of the distance error computation on two selected workpieces in reactive path planning experiments.

To measure path planning performance, we utilize three metrics: the proportion of trajectories where an agent achieves the target (*reached%*), and the proportion of trajectories where a collision occurs (*collision%*). When a collision happens, the sealing agent will stop. Otherwise, the sealing gun tip would deform and lead to imprecision of the TCP transformation, which is not tolerated for high-precision sealing tasks. Therefore, we have *reached%* + *collision%* = 100%. Additionally, an RMSE between the generated trajectories and the ground truth is also considered measuring the sealing application performance. We collected over 100 trajectories for each workpiece and presented the results in Table 4.1. The performance of our neural RMP agent is close to *predictingcontrols* on training workpieces (flat and cylinder), with comparable high *reached%* and low *collision%*. However, for the unseen rectangular workpiece, the *predictingcontrols* is more likely to get stuck, with only performing 60% on reaching the goal points. On the contrary, our neural RMPs still keep the high *reached%* for the

Methods	Flat				Cylinder				Rectangular			
	Collision	Reached	RMSE	Time	Collision	Reached	RMSE	Time	Collision	Reached	RMSE	Time
Ref [19]	32%	68%	3.79	45.71	59%	41%	4.30	35.76	88%	12%	6.47	53.72
Controls	27%	73%	3.57	35.62	35%	65%	5.29	27.47	69%	31%	5.65	43.92
RMPs	4%	**96%**	**0.83**	**30.04**	5%	**95%**	**0.94**	**22.36**	7%	**93%**	**0.99**	**38.47**

Table 4: Statistics on the three sealing workpieces in reactive path planning experiments. In addition to predicted controls, our neural RMP is also compared with a classical adaptive path planning method [19], which also leverages the feature extractions in ROIs to adaptively replan the path. We evaluate selected approaches from four performance metrics: *Collision %*, *Reached %*, RMSE and running time.

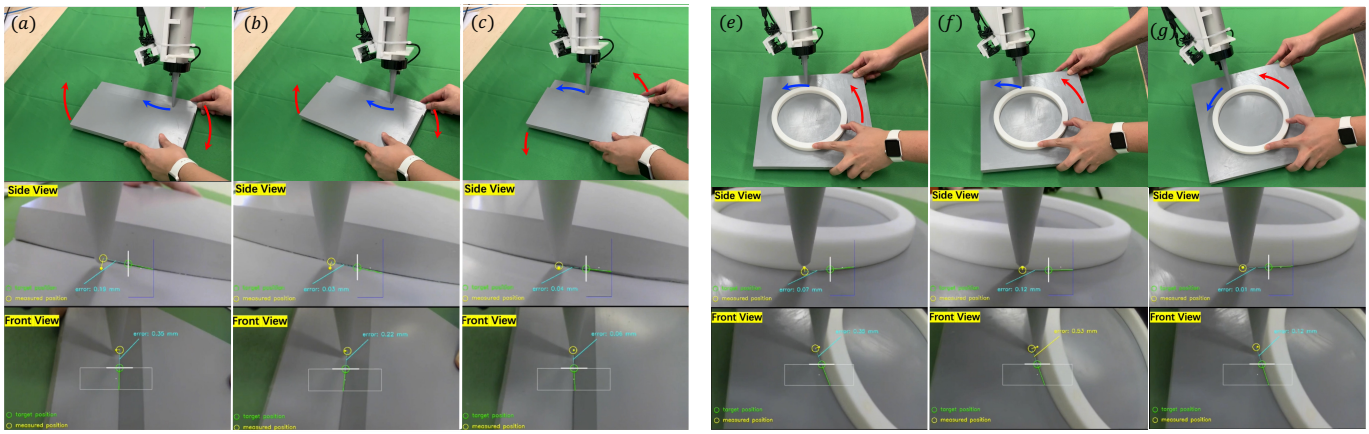

Fig. 15: Illustration of *Neural RMPs* for reactive path planning under dynamical environment disturbance. (a)-(c) and (e)-(g) show how *Neural RMPs* control the sealing gun to complete path planning when the sealing workpieces are rotating and translating.

goal point reaching, which shows that our neural RMPs generalize much better than predicting controls due to their explicit modeling of the sealing gun geometry and dynamics. In addition to the percentage of *reached%*, our neural RMPs keeps the RMSE within 1*mm*, which outperforms the *predicting controls* approach on all the sealing workpieces. Compared to predicting controls, the RMP representation is more succinct and less noisy, making it more resilient when a robot manipulator operates in tight locations where slight measurement mistakes in the geometry might result in failures.

Fig. 13 (a)-(g) show the top-down and side views of sampled trajectories on the three different sealing workpieces by *Neural Controls* and *Neural RMPs*. From the top-down views, the trajectories performed with *Neural Controls* are close to those with our *Neural RMPs*, but when checking on the side views, it's obvious that *Neural Controls* generates a trajectory with larger errors compared with our *Neural RMPs* model. Among the three workpieces, the last one is challenging because they require sharp turns that is unseen in the training data set. Also, the tight spaces have a low tolerance for controller models. *Neural Controls* tends to drift too much in this working environment and ends up hitting the workpiece. In comparison, *Neural RMPs* succeeded in all cases. Fig. 13 (h), (j), and (l) show the final real sealing experiments performed with our *Neural RMPs*. The corresponding front and side views visualize the real-time sealing tip position, attraction point, and edge detected in the ROIs. The *Neural RMPs* is able to process the attraction distance as small as possible in all the sealing cases. From the side views, the silicone sealant is evenly distributed on the sealing seam with the sealing gun moving under the guidance of *Neural RMPs* model. After sealing, a smoothing process is necessary to remove the redundant sealant to keep full contact with the moisture in the air to cure the sealing seam rapidly. As shown in Fig. 13 (i), (k), and (m), we also perform the smoothing process for each sealed workpiece, and the trajectories used for smoothing the silicone sealant are generated based on the previous sealing trajectories with an appropriate computed transformation matrix.

4.3. Reactive Path Planning

To evaluate the performance of our neural RMP in dynamic environment, a series of reactive path planning experiments is conducted. Similar to our above experiments, the ground truth trajectories are generated by calibrating the transformation between the position of the center of the AR code labeled on the workpieces and the CAD models. As shown in Fig. 14, the ground truth trajectories are marked with blue lines. Note that the RMSE to evaluate the performance of reactive path planning needs to be computed in a real-time manner since the ground truth is changing along with the environmental disturbance. The distance error is computed as the distance error between the sealing gun tip and the nearest point in the ground truth trajectories, which is marked as red lines in the figure. In addition to predicted controls, our neural RMP is also compared with a classical adaptive path planning method [19], which also leverages the feature extractions in ROIs to adaptively replan the path. To ensure a fair comparison, we replace the laser sensor with a commodity consumable depth sensor and slightly changed our implementation experimental setup in [25]. Similar to previous experiments, for each approach, we ran 100 tri-

als for each workpiece and reported the experimental quantitative in Table 4.1. Besides, we introduced a new metric of running time for each approach from the same starting point to the ending point under environmental disturbances. From the table, it can be observed that our approach achieves the highest *reached%*, lowest *collision%*, and RMSE. More importantly, our approach also achieves the shortest running time, which indicates highly acceptable efficiency. On the other hand, Ref [19] performs comparable results on the flat workpiece with the predicted controls approach, but it fails to handle cylinder and rectangular workpieces as this approach is not able to deal with complex dynamics of ROI features (e.g., dynamically changed curved path features). As for the *predicted controls* method, it fails to maintain the high *reached%* on flat and cylinder workpieces compared with its performance in Table 4.1, because the supervised approach is not able to build comprehensive and accurate connections between extracted features and motion control policies. For the unseen rectangular workpiece, it is more likely to get stuck around the corners compared to our *neural RMPs*.

Qualitative results of our neural RMPs can be found in Fig. 15, where the sealing workpieces are rotating and translating during the path planning process. The learned Riemannian metrics assign high costs to the deviating directions toward the sealing seam and low costs to the attraction points, as shown by the decreasing distance between the attraction point and the sealing gun tip. In addition, the learned Riemannian metrics try to reduce the sealing seam angles in both front and side views, but it is less obvious compared with reaching the attraction point. It is because the learned Riemannian metrics are trained with cost function on the basis of the RMSE of the ground truth, so the heading attraction point is more important in terms of reducing the RMSE compared to changing the direction. The same argument also applies to the predicted accelerations. Therefore, we can analyze the predicted RMPs to intuitively reason about the reactive behavior of the sealing gun tip, which would allow us to add appropriate RMPs to improve the reactive behavior of the sealing gun tip without having to retrain the network.

4.4. Limitations

Our neural RMPs model for reactive path planning holds an assumption that the sealing seam could be precisely represented by an edge. Although most of the sealing applications only work on commonly used seam types (e.g., straight line, structured curve), there are still some application scenarios where the sealing seam is so complicated that an edge representation is not able to describe it. In that case, a wider camera perspective and a more precise seam representation method are necessary. Additionally, our current pneumatic sealing system can only provide a consistent and stable airflow, which means we couldn't change the air pressure to control the silicone depositing speed. Therefore, the system is not able to ensure the deposited silicone sealant with an appropriate and uniform thickness (e.g., the sealing gun deposits excess silicone sealant around the corner on the rectangular workpiece), which is important for sealing quality performance. In the future, we could introduce an auto-sealant regulation system into our current robotic sealing system. By designing a vision-based algorithm to add the feedback signals of the silicone sealant, if the silicone sealant deposited in the groove is becoming thicker, then we will reduce the air pressure to regulate the sealant depositing speed. Since sealant thickness is of great importance to the sealing performance, we will incorporate this vision-based sealant regulation system design into our future work.

5. Conclusion

In this paper, we present a novel two-camera-based neural path planning framework for robotic sealing tasks. Specifically, to cope with the low-SNR issue of cheap cameras, on the basis of fractional-order differentiation operation, a robust sealing seam detection pipeline is proposed first to extract the edge-related features in predefined ROIs. Then, to generate desired sealing behaviors, we leverage the RMPs to model the geometry and dynamics of the sealing gun tip and its interaction with the sealing seams above the workpieces. Compared with directly predicting the control commands, our path planning approach achieves more generalizable on the unseen workpieces and in the dynamic environment under human disturbances. Experiments of physical sealing on various workpieces with different joints have validated the effectiveness of the proposed framework. As our framework models the geometry and dynamics explicitly, we believe it has strong potential in image-based robot path planning, policy transfer, and agile robot maneuver. In the future, a high-level perspective path planning framework (like our previous research [25]) and an adjustable pneumatic sealing system that is able to control the sealant depositing will be considered. Additionally, we may introduce it into diverse robotic tasks, developing better neural architectures for RMPs learning [39].

References

[1] R. K. Flitney, Seals and sealing handbook, Elsevier, 2011.

[2] P. Maiolino, R. Woolley, D. Branson, P. Benardos, A. Popov, S. Ratchev, Flexible robot sealant dispensing cell using rgb-d sensor and off-line programming, Robotics and Computer-Integrated Manufacturing 48 (2017) 188–195.

[3] Z. M. Bi, C. Luo, Z. Miao, B. Zhang, W. Zhang, L. Wang, Safety assurance mechanisms of collaborative robotic systems in manufacturing, Robotics and Computer-Integrated Manufacturing 67 (2021) 102022.

[4] Z. Pan, J. Polden, N. Larkin, S. Van Duin, J. Norrish, Recent progress on programming methods for industrial robots, Robotics and Computer-Integrated Manufacturing 28 (2) (2012) 87–94.

[5] J. R. Panek, J. P. Cook, Construction sealants and adhesives, Vol. 71, John Wiley & Sons, 1992.

[6] C. Li, P. Zheng, S. Li, Y. Pang, C. K. Lee, Ar-assisted digital twin-enabled robot collaborative manufacturing system with human-in-the-loop, Robotics and Computer-Integrated Manufacturing 76 (2022) 102321.

[7] A. Rahimi, A. Mashak, Review on rubbers in medicine: natural, silicone and polyurethane rubbers, Plastics, rubber and composites 42 (6) (2013) 223–230.

[8] H. Golnabi, A. Asadpour, Design and application of industrial machine vision systems, Robotics and Computer-Integrated Manufacturing 23 (6) (2007) 630–637.

[9] H.-y. Shen, J. Wu, T. Lin, S.-b. Chen, Arc welding robot system with seam tracking and weld pool control based on passive vision, The International Journal of Advanced Manufacturing Technology 39 (7) (2008) 669–678.

[10] R. O. Buchal, D. B. Cherchas, F. Sassani, J. Duncan, Simulated off-line programming of welding robots, The International journal of robotics research 8 (3) (1989) 31–43.

[11] D. A. Forsyth, J. Ponce, Computer vision: a modern approach, prentice hall professional technical reference, 2002.
[12] J. Fan, P. Zheng, S. Li, Vision-based holistic scene understanding towards proactive human–robot collaboration, Robotics and Computer-Integrated Manufacturing 75 (2022) 102304.
[13] C. Kohrt, R. Stamp, A. Pipe, J. Kiely, G. Schiedermeier, An online robot trajectory planning and programming support system for industrial use, Robotics and Computer-Integrated Manufacturing 29 (1) (2013) 71–79.
[14] M. J. Tsai, H.-W. Lee, N.-J. Ann, Machine vision based path planning for a robotic golf club head welding system, Robotics and Computer-Integrated Manufacturing 27 (4) (2011) 843–849.
[15] H. Fang, S. Ong, A. Nee, Interactive robot trajectory planning and simulation using augmented reality, Robotics and Computer-Integrated Manufacturing 28 (2) (2012) 227–237.
[16] M. Kazemi, K. Gupta, M. Mehrandezh, Path-planning for visual servoing: A review and issues, Visual Servoing via Advanced Numerical Methods (2010) 189–207.
[17] A. Cherubini, D. Navarro-Alarcon, Sensor-based control for collaborative robots: Fundamentals, challenges, and opportunities, Frontiers in Neurorobotics (2021) 113.
[18] Y. Ding, W. Huang, R. Kovacevic, An on-line shape-matching weld seam tracking system, Robotics and Computer-Integrated Manufacturing 42 (2016) 103–112.
[19] R. Manorathna, P. Phairatt, P. Ogun, T. Widjanarko, M. Chamberlain, L. Justham, S. Marimuthu, M. R. Jackson, Feature extraction and tracking of a weld joint for adaptive robotic welding, in: 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV), IEEE, 2014, pp. 1368–1372.
[20] S. M. Ahmed, Y. Z. Tan, C. M. Chew, A. Al Mamun, F. S. Wong, Edge and corner detection for unorganized 3d point clouds with application to robotic welding, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 7350–7355.
[21] V. Patil, I. Patil, V. Kalaichelvi, R. Karthikeyan, Extraction of weld seam in 3d point clouds for real time welding using 5 dof robotic arm, in: 2019 5th International Conference on Control, Automation and Robotics (ICCAR), IEEE, 2019, pp. 727–733.
[22] L. Nele, E. Sarno, A. Keshari, An image acquisition system for real-time seam tracking, The International Journal of Advanced Manufacturing Technology 69 (9) (2013) 2099–2110.
[23] K. Zhang, M. Yan, T. Huang, J. Zheng, Z. Li, 3d reconstruction of complex spatial weld seam for autonomous welding by laser structured light scanning, Journal of Manufacturing Processes 39 (2019) 200–207.
[24] L. Yang, Y. Liu, J. Peng, Z. Liang, A novel system for off-line 3d seam extraction and path planning based on point cloud segmentation for arc welding robot, Robotics and Computer-Integrated Manufacturing 64 (2020) 101929.
[25] P. Zhou, R. Peng, M. Xu, V. Wu, D. Navarro-Alarcon, Path planning with automatic seam extraction over point cloud models for robotic arc welding, IEEE Robotics and Automation Letters 6 (3) (2021) 5002–5009.
[26] L. Zhang, Y. Xu, S. Du, W. Zhao, Z. Hou, S. Chen, Point cloud based three-dimensional reconstruction and identification of initial welding position, in: Transactions on Intelligent Welding Manufacturing, Springer, 2018, pp. 61–77.
[27] L. Jing, J. Fengshui, L. En, Rgb-d sensor-based auto path generation method for arc welding robot, in: 2016 Chinese control and decision conference (CCDC), IEEE, 2016, pp. 4390–4395.
[28] J. Qi, G. Ma, P. Zhou, H. Zhang, Y. Lyu, D. Navarro-Alarcon, Towards latent space based manipulation of elastic rods using autoencoder models and robust centerline extractions, Advanced Robotics 36 (3) (2022) 101–115.
[29] F. Belkhouche, Reactive path planning in a dynamic environment, IEEE Transactions on Robotics 25 (4) (2009) 902–911.
[30] J. Van Den Berg, D. Ferguson, J. Kuffner, Anytime path planning and replanning in dynamic environments, in: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., IEEE, 2006, pp. 2366–2371.
[31] D. A. Pomerleau, Alvinn: An autonomous land vehicle in a neural network, Advances in neural information processing systems 1 (1988).
[32] F. Codevilla, M. Müller, A. López, V. Koltun, A. Dosovitskiy, End-to-end driving via conditional imitation learning, in: 2018 IEEE international conference on robotics and automation (ICRA), IEEE, 2018, pp. 4693–4700.
[33] H. Xu, Y. Gao, F. Yu, T. Darrell, End-to-end learning of driving models from large-scale video datasets, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2174–2182.
[34] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, B. Boots, Agile autonomous driving using end-to-end deep imitation learning, arXiv preprint arXiv:1709.07174 (2017).
[35] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, D. Fox, Riemannian motion policies, arXiv preprint arXiv:1801.02854 (2018).
[36] A. Nandal, H. Gamboa-Rosales, A. Dhaka, J. M. Celaya-Padilla, J. I. Galvan-Tejada, C. E. Galvan-Tejada, F. J. Martinez-Ruiz, C. Guzman-Valdivia, Image edge detection using fractional calculus with feature and contrast enhancement, Circuits, Systems, and Signal Processing 37 (9) (2018) 3946–3972.
[37] L. Ding, A. Goshtasby, On the canny edge detector, Pattern recognition 34 (3) (2001) 721–725.
[38] O. R. Vincent, O. Folorunso, et al., A descriptive algorithm for sobel image edge detection, in: Proceedings of informing science & IT education conference (InSITE), Vol. 40, 2009, pp. 97–107.
[39] O. Zahra, S. Tolu, P. Zhou, A. Duan, D. Navarro-Alarcon, A bio-inspired mechanism for learning robot motion from mirrored human demonstrations, Frontiers in Neurorobotics 16 (2022).