

Learning to Shape by Grinding: Cutting-surface-aware Model-based Reinforcement Learning

Takumi Hachimine¹, Jun Morimoto^{2,3}, and Takamitsu Matsubara¹

Abstract—Object shaping by grinding is a crucial industrial process in which a rotating grinding belt removes material. Object-shape transition models are essential to achieving automation by robots; however, learning such a complex model that depends on process conditions is challenging because it requires a significant amount of data, and the irreversible nature of the removal process makes data collection expensive. This paper proposes a *cutting-surface-aware* Model-Based Reinforcement Learning (MBRL) method for robotic grinding. Our method employs a cutting-surface-aware model as the object’s shape transition model, which in turn is composed of a geometric cutting model and a cutting-surface-deviation model, based on the assumption that the robot action can specify the cutting surface made by the tool. Furthermore, according to the grinding resistance theory, the cutting-surface-deviation model does not require raw shape information, making the model’s dimensions smaller and easier to learn than a naive shape transition model directly mapping the shapes. Through evaluation and comparison by simulation and real robot experiments, we confirm that our MBRL method can achieve high data efficiency for learning object shaping by grinding and also provide generalization capability for initial and target shapes that differ from the training data.

I. INTRODUCTION

Object-shape manipulation is widely applied in industry and our daily life. Shaping methods include two major types: Deformation processing [1] shapes objects by bending or pressing, in the manner of forming a dough, while removal processing shapes objects by gradually removing unnecessary material from the base stock, such as cutting metal. As a common industrial removal process, this paper focuses on the grinding process shown in Fig. 1, in which material is removed by a rotating grinding belt, and explores the possibility of automating this process with a robot.

To enable the grinding process by a robot, modeling the object-shape transition is crucial; however, this is difficult because the removal process generates removal resistance depending on material hardness, abrasive grain size, and belt rotation speed [2], which drags on the robot’s end-effector and complicates the shape transition. Even if such environmental conditions are fixed, different processing positions and postures of the robot generate different removal

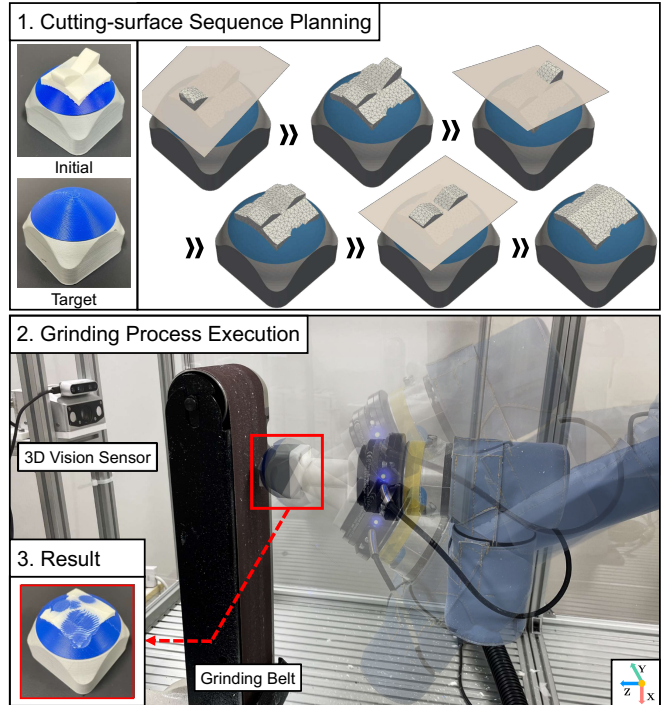


Fig. 1: Robotic object shaping by grinding. The robot shapes the object from the initial to the target shape, considering shape deviation by removal resistance. Robot actions are planned as a sequence of the cutting surface.

resistance, which affects removal shape and volume and thus still complicates the shape transition. Therefore, in this paper, as a first step toward automating the grinding process, we focus on the grinding process with various processing positions and postures of the robot as the process condition, while the environmental conditions are fixed.

Due to the complexity of the shape transition model in removal processes, it is reasonable to consider a data-driven model learning approach. Model-Based Reinforcement Learning (MBRL) has been successfully applied to various tasks as a framework that efficiently performs model learning and control planning [3], [4], [5]. In particular, we focus on the PILCO framework [5], which uses Gaussian process regression for model training. This is because it allows for the evaluation of the predictive uncertainty of the model, which is useful for grinding tasks that require avoiding robot motions with a high risk of damage due to model uncertainty. A naive model that takes the current shape and process conditions as inputs and learns a mapping that outputs the shape at the next step is, however, problematic because the shape representation (point cloud, depth image, etc.) is very high-dimensional and the shape

This work was supported by JST-Mirai Program Grant Number JP-MJMI21B1, Japan.

¹T. Hachimine and T. Matsubara are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Nara, Japan.

²J. Morimoto is with the Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan.

³J. Morimoto is also with the Brain Information Communication Research Laboratory Group (BICR), Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan.

transition depends on the process conditions. Therefore, a model learned with one particular initial and target shape cannot apply to different initial and target shapes, even for the same material is prone to generalization error. Therefore, an MBRL framework that satisfies 1) high data efficiency and 2) applicability of the learned models to different initial and target shapes is needed for autonomous object shaping by grinding.

To satisfy the above requirements, the removal process is performed by local surface contact between the tool surface (grinding belt) and the object. In other words, by interpreting that the shape transition is accomplished by the *cutting surface*, the shape transition can be represented by a) a geometric shape transition model with a cutting surface that ignores the effects of process conditions and b) a model that predicts such cutting-surface deviation that is dependent on process conditions. The former model can be commonly applied to a wide variety of shapes because the transitions are geometrically computable and require no learning. The latter model needs learning, but according to the grinding resistance theory [6], it does not require raw shape information, which reduces model dimensions and generalization capability.

With the above in mind, we propose a cutting-surface-aware MBRL method for object shaping by grinding. Our shape transition model, Cutting-Surface-Aware Model (CSAM), is composed of a) a Geometric Cutting Model (GCM) as a geometric shape transition model and b) a Cutting-Surface-Deviation Model (CSDM) that predicts the deviation of the cutting surface depending on the process conditions, based on the assumption that the robot action can specify the cutting surface made by the tool. Our MBRL iteratively optimizes both CSAM through learning CSDM and a cutting-surface sequence as robot actions through Model Predictive Control (MPC) for object shaping by grinding. We verified the performance of the proposed method in simulation and with real robot experiments. The results indicate that our method can achieve high data efficiency for learning object shaping by grinding and also provide generalization capability for initial and target shapes that differ from the training data. The following are this paper's main contributions:

- We proposed the first shape transition modeling method based on the cutting surface for grinding processes by a robot, taking into account the high dimensionality of the shape representation and dependence on process conditions.
- We proposed an MBRL method for learning a shape transition model and planning the optimal cutting-surface sequence with MPC.
- We verified the performance of the proposed method in simulation and real robot experiments.

II. RELATED WORK

A. Robotic Object-shape Manipulation

There have been many studies on shape manipulation tasks using robots. However, conventional methods focus on the

deformation process, such as ropes [7], [8], cables [9], [10], and fabrics [11], [12]. In contrast, studies on the removal process have been limited, perhaps due to the difficulty of modeling the shape transition depending on the process conditions. Methods for shape splitting have been studied in the Computer Vision field [13], but limited to geometrical computation algorithms. A related study of removal processing is cutting food items with a knife. Heiden *et al.* developed a differentiable cutting simulator to reproduce the force applied to a knife when cutting food [14]. By optimizing simulation parameters based on actual measurements, they constructed a simulator for various ingredients. Another related study involved polishing tasks. In order to improve surface properties by polishing, Yang *et al.* proposed a framework that collects process conditions to obtain the optimal parameters that satisfy the desired surface property using a genetic algorithm [2].

In any case, these studies did not construct a shape transition model due to the complexity of the shape transition by the removal process. Therefore, the conventional method is an adaptive method that optimizes parameters (e.g., robot speed and material properties) within an arbitrarily defined trajectory. In contrast, our proposed method can model the shape transition based on the cutting surface and plan robot action using MPC.

B. Object-shape Manipulation with Reinforcement Learning

Several studies have been conducted using both model-free RL and model-based RL. These methods may be preferable for grinding processes because there is no risk of injury or physical burden on the human demonstrator compared to such imitation learning approaches as Behavior Cloning [15] that use demonstration data. Model-free RL is effective for learning policy directly from high-dimensional inputs; however, it requires a large amount of data collection to obtain optimal control policies. Physics simulators for deformation processing [11], [16] are often used to collect training data because real-world experiments are costly. A typical physics simulator for removal processes is Computer-Aided Manufacturing (CAM). CAM can calculate the removal resistance applied to the tool and the processing time by setting variables such as the tool path and processing parameters. Therefore, a very high-power and high-rigidity mechanism is required, which is not suitable for shape manipulation with a multi Degree of Freedom (DoF) robot. Furthermore, the finite element and particle methods are computationally expensive.

Consequently, we employ the MBRL framework for high data efficiency. In addition, by introducing CSAM, we only need to learn CSDM, which depends on process conditions, instead of the direct learning input-output shape-transition model. This approach allows certain generalization capabilities for different initial and target shapes with a reasonable amount of data collection.

III. PROPOSED METHOD

This section explains our proposed method. Section III-A describes the shape transition model called CSAM, Section

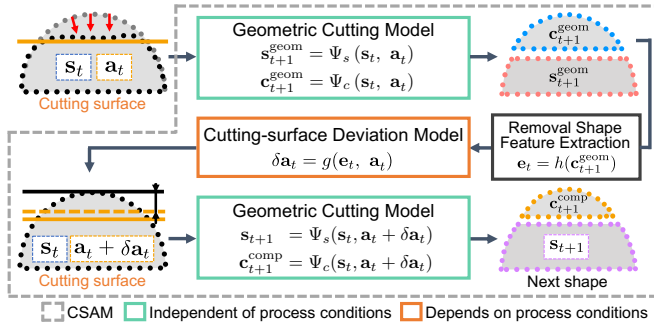


Fig. 2: Representation of removal process shape-transition model by Cutting-Surface-Aware Model (CSAM), which is composed of a) a Geometric Cutting Model (GCM) for geometrically splitting the shape by the cutting surface independent of the process conditions and b) a Cutting-Surface-Deviation Model (CSDM) for predicting the deviation of the cutting surface due to removal resistance.

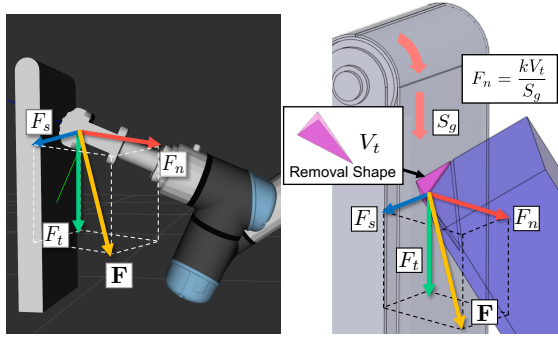


Fig. 3: Three component forces of grinding resistance on grinding belt

III-B discusses the planning of cutting-surface sequences by MPC, and Section III-C introduces our MBRL method that iteratively optimizes the CSDM and action plans.

A. Cutting-Surface-Aware Model (CSAM)

The proposed CSAM shown in Fig. 2 is composed of a) GCM, which geometrically splits the shape by the cutting surface independent of the process conditions, and b) CSDM, which predicts the deviation of the cutting surface due to removal resistance. We denote $s_t \in \mathcal{S}$ as the state representing the object shape, $\mathbf{a}_t \in \mathcal{A}$ as the robot action, and t as the time step. In our experiments as shown later, we employed the point cloud $\mathcal{O}_t := [\mathbf{q}_{i,t}]_{i=1}^D \subseteq \mathbb{R}^3$ as the state, where D is the number of points and $\mathbf{q}_{i,t}$ is the i -th particle position. Here, we interpret the shape transition as caused by the cutting surface; therefore, the robot's action is defined as a specification of the cutting surface.

1) *Geometric Cutting Model (GCM)*: We assume that a cutting surface (robot action) \mathbf{a}_t geometrically splits the current shape s_t into the next time-step shape s_{t+1}^{geom} and removal shape c_{t+1}^{geom} . Shape deformation by the cutting surface is defined as the following geometric cutting functions $\Psi(\cdot)$:

$$s_{t+1}^{geom} = \Psi_s(s_t, \mathbf{a}_t), c_{t+1}^{geom} = \Psi_c(s_t, \mathbf{a}_t). \quad (1)$$

These functions only geometrically split the shape by the cutting surface. Shape splitting is tractable because it is based on collision detection between the cutting surface and the shape [17], [18], in which computational cost is linear to the number of point clouds D .

2) *Cutting-Surface Deviation Model (CSDM)*: In actual removal processing, removal resistance occurs between the robot and the object, depending on the process conditions. In particular, the removal resistance generated by the grinding is called grinding resistance $\mathbf{F} = [F_t, F_n, F_s]^T$ as illustrated in Fig. 3. This resistance operates in 1) a tangential direction to the grinding belt F_t , 2) a vertical direction to the grinding belt surface F_n , and 3) parallel directions to the circumference of the grinding belt F_s .

Although the analytical formulation of grinding resistance is difficult due to the complexity of the physical grinding process, experimental results show that $F_n = kV_t/S_g$ and $F_t = \lambda kV_t/S_g$ [6], [19]. Both F_n and F_t are proportional to the removal volume V_t and inversely proportional to the rotational speed of the grinding belt S_g , where k is the specific grinding energy and λ is the ratio of F_n/F_t . In practice, the grinding resistance does not match the experimental formula owing to various other effects (e.g., friction heat, clogging of the grinding belt, the other force F_s).

With this grinding resistance theory $F_n = kV_t/S_g$, $F_t = \lambda kV_t/S_g$, we construct CSDM $g(\cdot)$, using a data-driven approach, to predict the deviation of the cutting surface due to grinding resistance, which depends on process conditions. The model takes the removal shape features (e.g., volume, size) from the removal shape c_{t+1}^{geom} based on grinding resistance theory. By using the CSDM $g(\cdot)$, the deviation of the cutting surface δa_t can be expressed as follows:

$$\delta a_t = g(e_t, \mathbf{a}_t), e_t = h(c_{t+1}^{geom}). \quad (2)$$

The deformed shape s_{t+1} can be expressed using the geometric cutting function $\Psi_s(\cdot)$ and the cutting-surface deviation function $g(\cdot)$:

$$\begin{aligned} s_{t+1} &= \Psi_s(s_t, \mathbf{a}_t + \delta a_t) = \Psi_s(s_t, \mathbf{a}_t + g(e_t, \mathbf{a}_t)) \\ &= \Psi_s(s_t, \mathbf{a}_t + g(h(\Psi_c(s_t, \mathbf{a}_t)), \mathbf{a}_t)) \\ &\triangleq f(s_t, \mathbf{a}_t). \end{aligned} \quad (3)$$

In summary, by representing the shape-transition model as GCM and CSDM, it can be expressed by the definition of the cutting surface at each time step. Note that CSDM does not use the removal shape directly but only the extracted features. Therefore, the learned CSDM is expected to have the ability to generalize across different initial and target shapes.

3) *CSDM training with Gaussian Process*: In this study, Gaussian process (GP) regression [20] is used to learn the CSDM. The deviation of cutting surface δa_t depends on the extracted features of the removal shape e_t and robot action \mathbf{a}_t . Thus, We define $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^T$, $\mathbf{x}_n := [e_t, \mathbf{a}_t]$ as training inputs, and $\mathbf{Y} = [\delta a_1, \dots, \delta a_N]^T$ as training outputs, where N is data size.

Assuming the GP as prior distribution for the function $g(\cdot)$, the predicted mean $m(\mathbf{x}_*)$ and predicted variance $\sigma^2(\mathbf{x}_*)$ for the new input data \mathbf{x}_* are computed as follows: $p(g(\mathbf{x}_*) | \mathbf{X}, \mathbf{Y}) = \mathcal{N}(g(\mathbf{x}_*) | m(\mathbf{x}_*), \sigma^2(\mathbf{x}_*))$, $m(\mathbf{x}_*) = \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{Y}$, $\sigma^2(\mathbf{x}_*) = k_{**} - \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_*$, where

$\mathbf{k}_* = k(\mathbf{X}, \mathbf{x}_*)$, $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$, $k(\cdot, \cdot)$ is kernel function, and \mathbf{K} is a kernel gram matrix that has $K_{i,j} = k(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$. Therefore, the deviation of cutting surface δa_* is predicted as GP mean $m(\mathbf{x}_*)$, and $\sigma^2(\mathbf{x}_*)$ is the variance of δa_* , which is used as the cost function. Grinding resistance theory suggests removal resistance is proportional to removal volume at one time; hence, such a smooth kernel as Squared Exponential (SE) kernel can be used. If the output values are multidimensional, the prediction is executed by preparing multiple GPs for the number of dimensions.

B. Cutting-surface Sequence Planning with MPC

With the learned shape-transition model, the optimal robot-action sequence from the current shape to the target shape can be obtained using an MPC framework, which can be formulated as follows:

$$\left. \begin{aligned} [\mathbf{a}_{t+1}^*, \dots, \mathbf{a}_{t+H}^*] &= \arg \min_{\mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+H}} \frac{1}{H} \sum_{\tau=t+1}^{t+H} C(\hat{\mathbf{s}}_\tau, \mathbf{a}_\tau), \\ \text{subject to } \mathbf{s}_{t+1} &= f(\mathbf{s}_t, \mathbf{a}_t), \end{aligned} \right\} \quad (4)$$

where H is the planning horizon, τ is the planning index, $t = 1, 2, \dots, T$ is the task horizon, and $\hat{\mathbf{s}}_\tau$ is the shape predicted by CSAM. The cost function $C(\hat{\mathbf{s}}_\tau, \mathbf{a}_\tau)$ for object shaping can be defined as:

$$\begin{aligned} C(\hat{\mathbf{s}}_\tau, \mathbf{a}_\tau) &= \underbrace{d_{\text{CD}}(\hat{\mathbf{s}}_\tau, \mathbf{s}_{\text{target}})}_{\text{Shape error cost}} + \underbrace{\phi(\hat{\mathbf{s}}_\tau, \mathbf{s}_{\text{target}})}_{\text{Over cutting cost}} \\ &+ \underbrace{\eta \frac{1}{M} \sum_{j=1}^M \sigma_j^2([h(\Psi_c(\hat{\mathbf{s}}_\tau, \mathbf{a}_\tau)), \mathbf{a}_\tau])}_{\text{Variance cost}}, \end{aligned} \quad (5)$$

where $[h(\Psi_c(\hat{\mathbf{s}}_\tau, \mathbf{a}_\tau)), \mathbf{a}_\tau] = [\mathbf{e}_\tau, \mathbf{a}_\tau] = \mathbf{x}_*$, $d_{\text{CD}}(\mathbf{s}_1, \mathbf{s}_2)$ is Chamfer discrepancy [21], which evaluates the error between two point-cloud data sets:

$$d_{\text{CD}}(\mathbf{s}_1, \mathbf{s}_2) = \frac{1}{|\mathbf{s}_1|} \sum_{\mathbf{p} \in \mathbf{s}_1} \min_{\mathbf{q} \in \mathbf{s}_2} \|\mathbf{p} - \mathbf{q}\|_2^2 + \frac{1}{|\mathbf{s}_2|} \sum_{\mathbf{q} \in \mathbf{s}_2} \min_{\mathbf{p} \in \mathbf{s}_1} \|\mathbf{p} - \mathbf{q}\|_2^2. \quad (6)$$

In Eq. 5, the shape error cost refers to error between the target shape $\mathbf{s}_{\text{target}}$ and the predicted shape $\hat{\mathbf{s}}_\tau$. The over-cutting cost is a penalty for the state of over-cutting the target shape. The variance cost refers to avoiding uncertain actions, $\sigma_j^2(\cdot)$ is the predicted variance of the GP, M is the number of predicted dimensions, and η is the coefficient.

C. Iterative Optimization of CSDM and Cutting-Surface Sequence Planning by MBRL

Finally, we formulate the iterative optimization of CSDM and MPC action planning (and execution) as an MBRL framework for high data efficiency as shown in Fig. 4 and Algorithm.1. A total of T steps are defined as one learning episode of the RL process, and the total number of learning times is denoted as N_{episode} . In advance, we collected initial data using only GCM with N_{init} random actions and trained CSDM (L5-11). The proposed framework follows an iterative procedure: 1) Execute MPC using CSAM. MPC minimizes

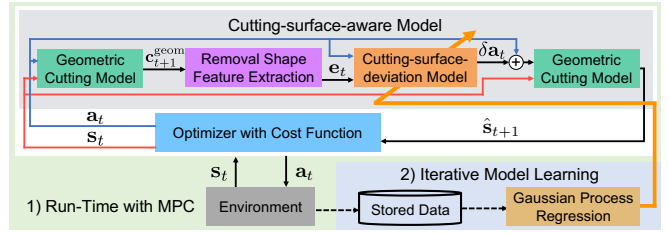


Fig. 4: Our MBRL framework of the object-shaping planner for the removal process. The proposed method iteratively optimizes 1) the planning (and execution) of the cutting-surface sequences by MPC using the cutting-surface-aware model and 2) the training and updating of the cutting-surface-deviation model with collected data from MPC run-time.

Algorithm 1: Cutting-surface Aware MBRL

```

1 Input: RL episode  $N_{\text{episode}}$ , Task horizon  $T$ , Predict Horizon  $H$ , Cost function  $C(\cdot)$ 
2 Initialization: CSDM dataset  $\mathcal{D} \leftarrow \{\mathbf{X} = \{\}, \mathbf{Y} = \{\}\}$ 
3 # Generate random samples with random actions
4 Set New Object()
5 for  $i = 1, \dots, N_{\text{init}}$  do
6    $\mathbf{s}_i = \text{Get state}()$ 
7    $\mathbf{a}_i = \text{RandomActions}()$ 
8   Execute( $\mathbf{a}_i$ )
9    $\mathbf{s}_{i+1} = \text{Get state}()$ 
10   $\mathbf{x}_* \leftarrow [\mathbf{e}_i, \mathbf{a}_i]$ ,  $\mathbf{y}_* \leftarrow [\delta \mathbf{a}_i]$ 
11   $\mathbf{X} = \{\mathbf{x}, \mathbf{x}_*\}$ ,  $\mathbf{Y} = \{\mathbf{y}, \mathbf{y}_*\}$ 
12  model = train CSDM( $\mathcal{D}$ )
13 # Model-based RL with MPC
14 for  $i = 1, \dots, N_{\text{episode}}$  do  $\triangleright$ Iterative Model Learning
15   Set New Object()
16   for  $t = 1, \dots, T$  do  $\triangleright$ Run-Time with MPC
17      $\mathbf{s}_t = \text{Get state}()$ 
18      $\mathbf{a}_t = \text{ActionSearch}(\mathbf{s}_t, H, \text{CSAM}, C(\cdot))$ 
19     Execute( $\mathbf{a}_t$ )
20      $\mathbf{s}_{t+1} = \text{Get state}()$ 
21      $\mathbf{x}_* \leftarrow [\mathbf{e}_t, \mathbf{a}_t]$ ,  $\mathbf{y}_* \leftarrow [\delta \mathbf{a}_t]$ 
22      $\mathbf{X} = \{\mathbf{x}, \mathbf{x}_*\}$ ,  $\mathbf{Y} = \{\mathbf{y}, \mathbf{y}_*\}$ 
23   model = train CSDM( $\mathcal{D}$ )

```

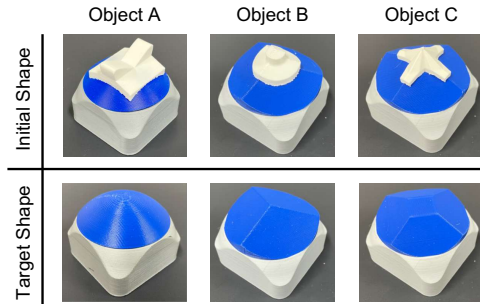


Fig. 5: Initial and target shapes for experiments

the cost function and selects the next action (cutting surface) based on the observed shape in the task horizon t . At each action step, we collect data to update CSDM (L16-22). 2) Train and update CSDM at the end of the episode (L23). By repeating this process N_{episode} times, the proposed framework learns CSDM and obtains the optimal cutting-surface sequence.

IV. ROBOTIC ROUGH GRINDING ENVIRONMENTS

This section describes our robotic rough grinding environment as shown in Fig. 1. The grinding belt is fixed in the environment, and the object is mounted on the 6-DoF robot's

end-effector. Fig. 5 shows the three different initial and target shapes of the objects used for the experiment. These were all fabricated by a 3D printer. The gray part (bottom) is the base, the blue part (middle) is the target shape, and the white part (top) is the material to be removed by grinding.

The robot action \mathbf{a}_t manipulates the coordinates of the object around the work origin, which is an arbitrary distance away from the grinding belt, as equivalent to specifying the cutting surface of the object. In the grinding process using the grinding belt, we assume that the belt surface is sufficiently wide. The 2-DoF of parallel translation and the 1-DoF rotation around a perpendicular axis can be ignored. As a result, robot action becomes $\mathbf{a}_t := [\text{roll}_t, \text{pitch}_t, z_t] \in \mathbb{R}^3$. Since our MPC plans cutting surface at each time step, the planned cutting surface is executed by the following controllers: 1) Transition from the work origin to the target frame, 2) Holding the target frame for an arbitrary time, and 3) Returning to the work origin from the target frame.

V. SIMULATION EXPERIMENT

We evaluated the proposed method with a simulator having grinding dynamics. This experiment has three objectives: The first is to confirm the learning performance of our MBRL for grinding. The second is to evaluate the performance of optimized action sequences for grinding with the learned CSDM. We compared five comparison methods; details are described in Section V-B. The third objective is to verify the applicability of the learned CSDM to different initial and target shapes. For this purpose, the CSDM learned by Object A was applied to process Object B and to Object C and then evaluated. The details of MBRL and MPC parameters used in experiments are on our project page¹.

A. Simulation Environments

The initial and target shapes, modeling the same ones in Fig. 5, are created by CAD and converted to the point cloud. Open3D [22] and Pyvista [23] are used for rendering.

To derive the deviation of the cutting surface, we implemented a virtual grinding resistance model, which depends on the robot action and removal volume at each time step. In the actual grinding process, if the removal volume exceeds the physical limit, the grinding resistance is massively increased. These conditions may lead to the destruction of the robot and the object. Consequently, the robot interrupts the action when grinding resistance is applied above a certain threshold to the end-effector. To replicate this mechanism, the simulator sets a massive deviation of the cutting surface when the removal volume at one time step exceeds a threshold.

B. Comparison Methods

To confirm the effectiveness of the proposed method, we prepared comparison methods as follows. **Random**: selects actions randomly at each time step from a uniform distribution. If the selected action exceeds the target shape, it is chosen again. **Geometric**: performs MPC using only the GCM. **Direct Shape Deformation Regression (DirectReg)**:

performs MPC using a transition model that directly predicts the shape at the next step based on the current shape and robot action. The transition model is constructed with Point Cloud Auto Encoder (PCAE) using PointNet [24]. **MFRL-SAC**: Performs Model-free RL by Soft Actor-Critic [25]. **Proposed-GT**: MPC performs with ground truth cutting-surface deviation to confirm the upper bound of task performance in the proposed method.

C. MBRL and MPC Settings

CSDM learns 1-axis deviation $y_i = [\delta a_t]$ of the cutting surface from input $\mathbf{x}_i := [\mathbf{e}_t, \mathbf{a}_t] = [V_t^{\text{geom}}, \text{roll}_t] \in \mathbb{R}^2$, where V_t^{geom} is the removal volume, which is the sum of the Euclidean distance between the removal shape $\mathbf{c}_{t+1}^{\text{geom}}$ and the cutting surface divided by the number of removal shape point clouds. We randomly collected 15 cutting surfaces with a removal volume of $[0.0 \sim 4.0]$ as initial data to train CSDM. In training CSDM, we excluded from the training data that data obtained when the robot action was interrupted due to the removal volume exceeding a threshold value. The ARD squared exponential kernel [20] is used as the GP kernel for CSDM. MPC uses the Random Shooting method [4], [26] to minimize the cost function. The over-cutting cost of Eq. 5 returns 1000 if the predicted shape exceeds the target shape, and 0 otherwise. The coefficient of variance cost η is set by the following equation:

$$\eta = \alpha \left[\frac{d_{\text{CD}}(\mathbf{s}_t, \mathbf{s}_{\text{target}})}{d_{\text{CD}}(\mathbf{s}_{L,\text{init}}, \mathbf{s}_{L,\text{target}})} \right]^\beta, \quad (7)$$

where α is the scale parameter of variance cost and β is the reduction factor based on the ratio of the initial shape error to the current shape error of the shape \mathbf{s}_L used in training. The variance cost is affected by measurement noise in the real world, and thus it is decreased according to the ratio of the current shape error.

D. Simulation Results

1) *Learning Performance of Proposed MBRL*: Fig. 6 shows the learning curves of MBRL for object A under different weights α of the variance cost. In the following experiment, the shape error is calculated by Eq. 6. Note that Eq. 6 is calculated with the Euclidean distance between the nearest neighbor points of two-point clouds, hence it is rare for the shape error to be zero unless two-point clouds are perfectly matched. These results confirm the ability of the proposed MBRL to optimize cutting-surface sequences with reduced shape error by setting the variance cost appropriately.

2) *Evaluation with Comparison Methods*: Fig. 7 (a) shows the task performance of each method. **Proposed** uses each of the last two episodes of CSDM trained with three different initial data. **DirectReg-S**, **DirectReg-M**, and **DirectReg-L** were trained with 4.5K, 91K, and 910K step data, respectively. The Reference Line is defined at 15 [%] above the average final shape error processed by **Proposed-GT**, for the indicator of task progress. The graph shows that **Proposed** achieves the closest shape error to **Proposed-GT**.

¹<https://t-hachimine.github.io/csambml/>

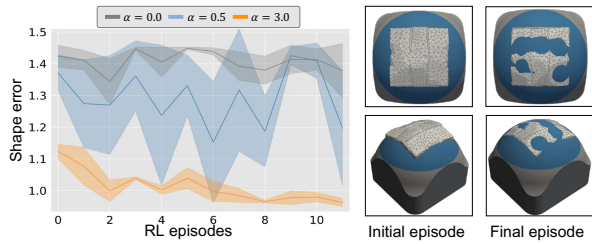


Fig. 6: MBRL results for Object A in simulation. **Left:** Learning curves, with mean value and standard deviation of three different initial data. **Right:** Rendered shape before and after learning ($\alpha = 3.0$).

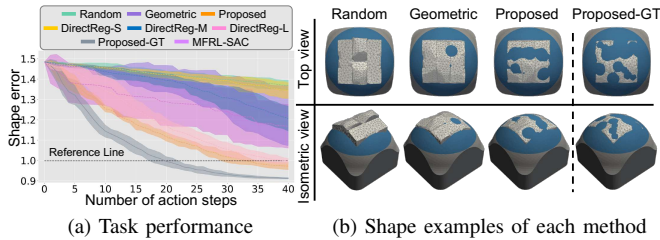


Fig. 7: Results of simulation experiments for Object A. (a) Task performance comparison for each method. (b) Rendered image of the shape at $t = 40$.

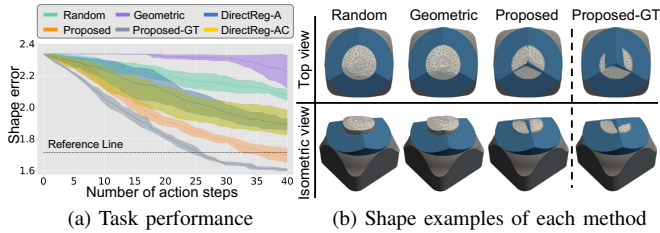


Fig. 8: Results of simulation experiments for Object B. (a) Task performance comparison for each method. (b) Rendered image of the shape at $t = 40$.

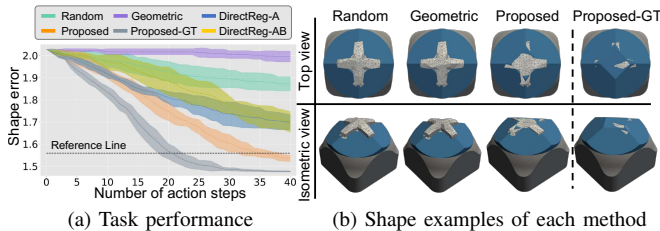


Fig. 9: Results of simulation experiments for Object C. (a) Task performance comparison for each method. (b) Rendered image of the shape at $t = 40$.

Geometric does not incorporate the deviation of the cutting surface by grinding resistance, and thus the task performance is low. In the case of **DeltaReg**, performance improves with increasing training data, but we need to collect more than 910K step data. Task performance of **MFRL-SAC** trained with 91K step data is comparable to **DirectReg-M**. On the other hand, **Proposed** achieves the highest performance with only 0.5K step data. These results confirm that the proposed method is more sample-efficient than model-free RL and direct shape prediction methods. Fig. 7 (b) shows an example of the shape after processing.

TABLE I shows the comparison of shape prediction error rate $|d_{CD}(\hat{s}_t, s_{target}) - d_{CD}(s_t, s_{target})| / d_{CD}(s_t, s_{target}) \times 100$ and deviation of the cutting surface at the first step $t = 1$ for six times of each method, to evaluate it from the same shape state. **Proposed** has a lower shape prediction error rate

TABLE I: Comparison of shape prediction and cutting surface deviation for Object A at $t = 1$ in simulation experiments. Here, *, **, † and ‡ denotes $p < 0.05$ on paired t-test.

Geometric	DirectReg-S	DirectReg-M	DirectReg-L	Proposed
Shape prediction error rate [%]				
$16.6 \pm 2.7^*$	$25.1 \pm 2.5^*$	$20.0 \pm 2.3^\dagger$	$13.1 \pm 2.2^\ddagger$	$0.16 \pm 0.3^{*,*,\dagger,\ddagger}$
Cutting surface deviation [mm]				
$7.0 \pm 1.8^*$	$5.2 \pm 2.2^*$	$5.2 \pm 2.7^\dagger$	3.5 ± 1.8	$2.1 \pm 0.76^{*,*,\dagger}$

and cutting-surface deviation than **Geometric**. The results suggest that learning CSDM improves the shape prediction accuracy of CSAM and allows the selection of actions with small cutting-surface deviation.

3) *Shape Generalization Performance:* We conducted six experiments on Objects B and C using CSDM learned for Object A. **Proposed** uses a final-episode CSDM model that was trained with three different initial data for Object A, and we conducted experiments on each twice. Fig. 8 and Fig. 9 show comparisons of task performance for Object B and Object C, respectively. **DirectReg-A** was trained with 91K step data using Object A, and **DirectReg-AB** and **AC** were trained with a total of 182K step data using Objects A and B, and Objects A and C, respectively. From Fig. 8 and Fig. 9, it can be seen that **DeltaReg** task performance is lower than **Proposed**. Therefore, a transition model with direct shape input and output would not provide sufficient shape generalization performance even if trained with multiple objects.

VI. REAL ROBOT EXPERIMENT

A. Construction of Robotic Rough Grinding System

We use a 6-DoF robot (UR5e) equipped with a force and torque sensor. In addition, a 3D vision sensor (YCAM3D-10M) is installed in the system to capture the object's shape. The robot moves in front of the 3D vision sensor and takes a point cloud. The belt grinder (BDS-100) is equipped with a grinding belt having a grain size of WA#80. Details of our experimental system are on our project page¹.

A hybrid position-force controller is implemented in the robot to prevent damage by grinding resistance. Furthermore, the robot has a protection control that interrupts the action when a force or torque over the threshold value is applied to the robot's end-effector. This protection control is adopted to prevent damage to the robot when large grinding resistance is suddenly applied.

B. MBRL and MPC Settings

CSDM learns 1-axis deviation $y_i = [\delta a_t]$ of the cutting surface from input $\mathbf{x}_i := [\mathbf{e}_t, \mathbf{a}_t] = [V_t^{\text{geom}}, w_t, h_t, d_t, \text{roll}_t, \text{pitch}_t] \in \mathbb{R}^6$, where w_t, h_t, d_t is the 3D-bounding box of the removal shape. In the actual grinding process, the grinding resistance depends on the contact area to the grinding belt, even if the removal volume at the time step is the same. Therefore, the input dimension of CSDM is extended. We randomly collected 60 cutting surfaces with a removal volume of $[0.1 \sim 2.0]$ as initial data to train

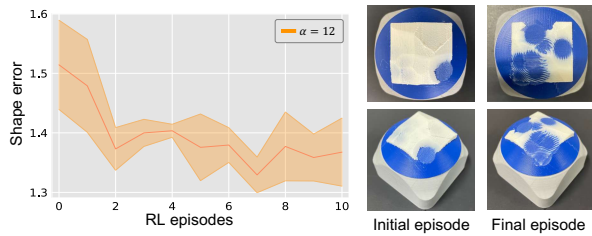


Fig. 10: MBRL results for Object A in robot experiments. **Left:** Learning curves, with mean value and standard deviation of three different initial data. **Right:** Captured shape before and after learning ($\alpha = 12.0$).

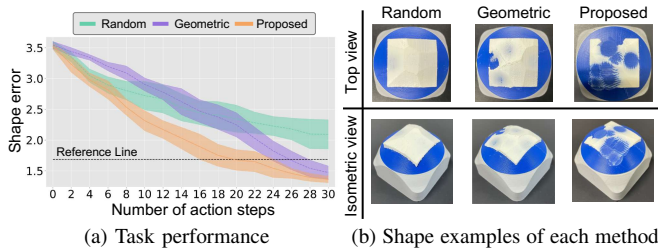


Fig. 11: Results of robot experiments for Object A. (a) Task performance comparison for each method. (b) Captured image of the shape at $t = 30$.

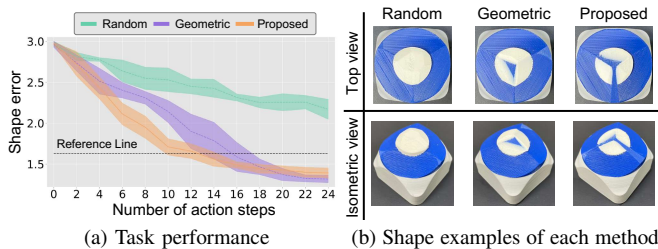


Fig. 12: Results of robot experiments for Object B. (a) Task performance comparison for each method. (b) Captured image of the shape at $t = 14$.

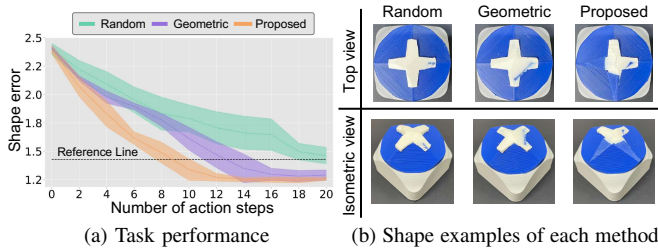


Fig. 13: Results of robot experiments for Object C. (a) Task performance comparison for each method. (b) Captured image of the shape at $t = 10$.

TABLE II: Comparison of the number of time steps to achieve Reference Line. Here, * denotes $p < 0.05$ on paired t-test.

	Geometric	Proposed
Object A	$27.3 \pm 1.9^*$	$20.0 \pm 3.1^*$
Object B	$16.7 \pm 2.5^*$	$13.0 \pm 2.0^*$
Object C	$14.0 \pm 2.3^*$	$9.7 \pm 1.4^*$

CSDM. In training CSDM, the data obtained when the robot action was interrupted by protective control were excluded. The MPC cost function follows Eq. 5 and its minimization method is the same as the simulation.

C. Results

1) *Learning Performance of Proposed MBRL:* Fig. 10 shows the learning curve of our MBRL for Object A. These results confirm that, similar to the simulation results, the

TABLE III: Comparison of shape prediction and cutting surface deviation for Object A at $t = 1$ in robot experiments. Here, * denotes $p < 0.05$ on paired t-test.

	Geometric	Proposed
Shape prediction error rate [%]	$27.0 \pm 5.37^*$	$5.10 \pm 2.32^*$
Cutting surface deviation [mm]	$6.85 \pm 1.30^*$	$1.37 \pm 1.86^*$

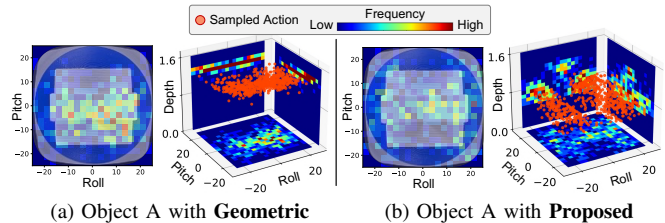


Fig. 14: Distribution of cutting surfaces sampled 600 times for the first step of Object A

proposed MBRL can optimize the cutting-surface sequence with reduced shape error in a real environment.

2) *Evaluation with Comparison Methods:* Fig. 11 (a) shows the task performance of each method. **Proposed** uses each of the last two episodes of CSDM trained with three different initial data. The settings of **Random** and **Geometric** are the same as in the Simulation. Reference Line is defined at 15 [%] above the average final shape error processed by **Proposed** because the ground truth cutting-surface deviation is unknown. In real robot experiments, the action space is limited compared to the simulation because the action is sometimes constrained based on the force sensor for safety reasons. Fig. 11 (a) shows that **Proposed** achieves Reference Line more quickly than **Geometric**. TABLE II compares the number of time steps to achieve the Reference Line between **Proposed** and **Geometric**. The paired t-test results show that the task performance of **Proposed** is significantly different from that of **Geometric**. Fig. 11 (b) shows an example of the shape after processing.

TABLE III shows the comparison of shape prediction error rate and deviation of the cutting surface at the first step $t = 1$ for six times of each method, to evaluate it from the same shape state. **Proposed** has a lower prediction shape error rate and cutting-surface deviation than **Geometric**. In addition, Fig. 14 shows the distribution of the cutting surface at $t = 1$. This distribution shows that **Proposed** selects the cutting surfaces with a small z_t (Depth) and large $roll_t$ (Roll) and $pitch_t$ (Pitch) compared to **Geometric**. These results suggest that learning CSDM improves the shape prediction accuracy of CSAM and allows the selection of actions with small cutting-surface deviation by optimizing the depth and angle of contact with the grinding belt.

3) *Shape Generalization Performance:* We conducted six experiments on Objects B and C using CSDM learned for Object A. **Proposed** uses a final-episode CSDM model trained with three different initial data for Object A, and we conduct experiments on each twice. TABLE II compares the number of time steps to achieve Reference Line. The paired t-test results show that the task performance of **Proposed** significantly differs from that of **Geometric**. Fig. 12 and

Fig. 13 show the task performance and the actual shape after processing by each method.

VII. DISCUSSION

This study focused on adaptation to the action-dependent process conditions in robotic grinding, assuming that environment-dependent process conditions are fixed. Therefore, we must retrain CSDM if the material hardness or grinding belt is changed. To achieve this, we will embed environment-dependent process conditions as parameters in the CSDM dataset, which is expected to provide, in future work, applicability to a wider variety of materials and grinding belts.

A limitation of the proposed method is that the robot motion is specified as a cutting surface, which may lead to inefficient processing for smooth shapes such as round surfaces. A possible solution to this problem is to extend the cost function to account for the continuity of the trajectory, but a multi-objective cost function may not be desirable from an optimization perspective. Therefore, learning a cost function that includes trajectory continuity based on human demonstrations [27] is an interesting future challenge. In addition, the trade-off between the computational cost of GCM and the quality of the shape representation could be a bottleneck for real-time action planning with MPC. We believe computational efficiency can be improved by implementing GCM with parallel computing.

VIII. CONCLUSION

We proposed the MBRL method for object shaping through removal processing by grinding. The key idea of the proposed method is to represent the shape-transition model by GCM, which is independent of process conditions, and CSDM, which only depends on process conditions, by introducing the cutting surface. Experiments were conducted on simulators and in the real world to evaluate the effectiveness of the proposed method. The experimental results confirmed that the proposed method achieves efficient learning with a reasonable amount of data and provides generalization capability for initial and target shapes that differ from the training data.

REFERENCES

- [1] J. Sanchez, J. Corrales, B. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.
- [2] H. Yang, C. Shih, Y. Lo, and F. Lian, "Zero-tuning grinding process methodology of cyber-physical robot system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4270–4275, 2020.
- [3] C. Matl and R. Bajcsy, "Deformable elasto-plastic object shaping using an elastic hand and model-based reinforcement learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3955–3962, 2021.
- [4] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, "Deep dynamics models for learning dexterous manipulation," in *Conference on Robot Learning*, pp. 1101–1112, 2020.
- [5] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning*, pp. 465–472, 2011.

- [6] J. Tang, J. Du, and Y. Chen, "Modeling and experimental study of grinding forces in surface grinding," *Journal of Materials Processing Technology*, vol. 209, no. 6, pp. 2847–2854, 2009.
- [7] T. Tang, C. Wang, and M. Tomizuka, "A framework for manipulating deformable linear objects by coherent point drift," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3426–3433, 2018.
- [8] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, M. Laskey, K. Stone, J. E. Gonzalez, and K. Goldberg, "Learning rope manipulation policies using dense object descriptors trained on synthetic depth data," in *2020 IEEE International Conference on Robotics and Automation*, pp. 9411–9418, 2020.
- [9] D. Sánchez, W. Wan, and K. Harada, "Tethered tool manipulation planning with cable maneuvering," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2777–2784, 2020.
- [10] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng, "Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks," in *IEEE International Conference on Robotics and Automation*, pp. 4568–4575, 2021.
- [11] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*, pp. 432–448, 2021.
- [12] Y. Tsurumine, Y. Cui, E. Uchibe, and T. Matsubara, "Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation," *Robotics and Autonomous Systems*, vol. 112, pp. 72–83, 2019.
- [13] E. Sifakis, K. Der, and R. Fedkiw, "Arbitrary cutting of deformable tetrahedralized objects," in *In Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 73–80, 2007.
- [14] E. Heiden, M. Macklin, Y. S. Narang, D. Fox, A. Garg, and F. Ramos, "DiSECT: A Differentiable Simulation Engine for Autonomous Robotic Cutting," in *Proceedings of Robotics: Science and Systems*, 2021.
- [15] D. A. Pomerleau, "Efficient Training of Artificial Neural Networks for Autonomous Navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.
- [16] Y. Hu, L. Anderson, T. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand, "DiffTaichi: Differentiable programming for physical simulation," *arXiv:1910.00935*, 2020.
- [17] S. Kumaressan, *Linear algebra: A geometric approach*. PHI Learning Pvt. Ltd., 2000.
- [18] F. Dunn and I. Parberry, *3D math primer for graphics and game development*. CRC Press, 2011.
- [19] W. B. Rowe, *Principles of modern grinding technology*. William Andrew, 2013.
- [20] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning, Cambridge, MA, USA: MIT Press, 2006.
- [21] T. Nguyen, Q. Pham, T. Le, T. Pham, N. Ho, and B. Hua, "Point-set distances for learning representations of 3d point clouds," *2021 IEEE/CVF International Conference on Computer Vision*, pp. 10458–10467, 2021.
- [22] Q. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.
- [23] C. B. Sullivan and A. Kaszynski, "PyVista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK)," *Journal of Open Source Software*, vol. 4, no. 37, p. 1450, 2019.
- [24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.
- [25] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, 2018.
- [26] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE international conference on robotics and automation*, pp. 7559–7566, 2018.
- [27] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel, "Adversarial motion priors make good substitutes for complex reward functions," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 25–32, 2022.