Generative Robotics: Self-Supervised Learning for Human-Robot Collaborative Creation

Peter Schaldenbrand

October 10, 2025
CMU-RI-TR-25-101

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania

Thesis Committee:
Prof. Jean Oh, *chair*
Prof. James McCann
Prof. Manuela Veloso
Prof. Ken Goldberg (University of California, Berkeley)

For the degree of Doctor of Philosophy in Robotics.

Abstract

Robot automation is generally welcomed for tasks that are dirty, dull, or dangerous, but with expanding robotic capabilities, robots are entering domains that are safe and enjoyable, such as creative industries. Although there is a widespread rejection of automation in creative fields, many people, from amateurs to professionals, would welcome supportive or collaborative creative tools. Supporting creative tasks is challenging with real-world robotics because there are limited relevant datasets, creative tasks are abstract and high-level, and real-world tools and materials are difficult to model and predict. Learning-based robotic intelligence is a promising method for creative support tools, but since the task is so complex, common approaches such as learning from demonstration would require too many samples and reinforcement learning may never converge. In this thesis, we introduce several self-supervised learning techniques to enable a robot to teach itself to support humans in the act of creativity.

We formalize robots that support people in the making of things from high-level goals in the real world as a new field, Generative Robotics. We introduce an approach for supporting 2D visual art-making with paintings and drawings along with 3D clay sculpture from a fixed perspective. Because there are no robotic datasets for collaborative painting and sculpting, we designed our approach to learn from small, self-generated datasets to learn real-world constraints and support collaborative interactions. This thesis contributes (1) a Real2Sim2Real technique that enables a robot to create complex dynamics models from small, self-generated datasets of actions, (2) a method for planning robotic actions for long-horizon tasks in a semantically aligned representation, and (3) a self-supervised learning framework to adapt pretrained models to be compatible with robots and produce collaborative goals. We show how self-supervised learning can enable model-based robot planning approaches to paint collaboratively with humans using various painting mediums. Lastly, we generalize our approach from the painting to the sculpting domain, demonstrating that our approach generalizes to new materials, tools, action representations, and state representations.

Dedicated to my family — Mum, Dad, Heinz, Liz, and Seamus —
the great gust of wind beneath my tattered wings.

Contents

List of Tables

List of Figures

I

Introduction

1 Introduction

> *Medicine, law, business, engineering, these are all noble pursuits and necessary to sustain life.*
>
> *But poetry, beauty, romance, love, these are what we stay alive for.*

John Keating, Dead Poet's Society, 1989

Robotic automation is excellent and welcome for tasks that are dirty, dangerous, or dull, but what about creative acts which are fun and fulfilling? With a recent boom in artificial intelligence (AI) capabilities such as text-to-image synthesis, artists fear losing employment and art viewers fear a degradation in quality of work [1]. This thesis concerns tasks where robots making things, however, we do not argue that robots should automate these tasks. Instead, robots can support artists in their work through collaboration, inspiration, and motivation, which many artists desire [2]. In the process of developing systems for artistic tasks, we uncover extraordinarily difficult technical challenges for robotics. In this thesis we introduce self-supervised learning techniques to tackle some fundamental challenges in robotics while working in the painting and sculpting domains.

1.1 Generative Robotics

The domain of this thesis is Generative Robotics; a field that is not necessarily novel, but we formalize in this document. Generative robots are robots that can support the making of things from high-level goals of a human user in the real world. There are three components of Generative Robotics that separate it from existing fields, such as manufacturing or Generative AI: Real-world constraints, high-level goals, and human-robot collaboration.

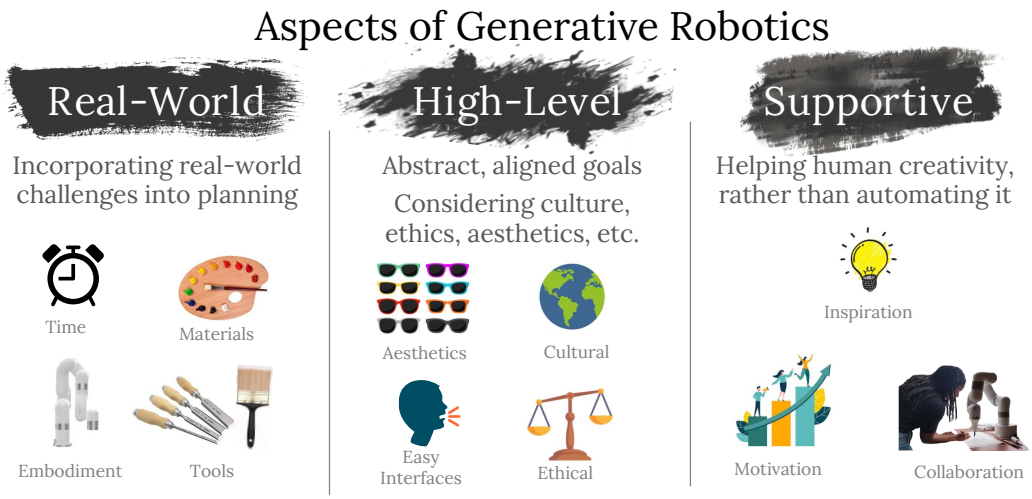

Figure 1.1: Aspects of Generative Robotics. There are three aspects of Generative Robotics that define it as a field and distinguish it from other robotics fields such as manufacturing or Generative AI.

Definition 1.1.1 (Generative Robotics)**.** Robots that can support the making of things from a human user's high-level goals in the real world.

1.1.1 Real World Constraints

The real world is full of constraints and challenges that make it difficult to physically make things. Materials are noisy and unpredictable, tools are challenging to control, and skillsets may be too limited to craft something complex. In most manufacturing settings with robotics, these real-world factors are controlled as much as possible. For example, a printer uses precise lasers with ink that flows through super controlled channels and paper that is cut into regular shapes and is completely blank. This is in contrast to a painter who uses paint that is challenging to mix into desired colors, brushes that have limited precision for small details, and may be working from a partially completed painting rather than blank canvas. If robots are to truly support people in the real world, these factors should be embraced and incorporated into the planning of the robot rather than finding work-arounds for them. In this thesis, we show that real-world constraints can be learned through self-generated actions to improve the accuracy of dynamics models, which enables model-based planning to be realistic.

1.1.2 High Level Goals

Users of a Generative Robotics system may have a limited skillset, meaning they may need high-level interfaces for interacting with the system. Whereas a low-level goal may require familiarity with complex software such as Computer-Aided Design systems, a high-level goal may be specified using common skills, such as natural language and decision making. While high-level goals may enable easier interface for users, they require more complex intelligence on the robot's side. Fortunately, technology has vastly improved in recent years at the making of things especially from high-level goals (Definition 1.1.3). Text-to-Image generators [3, 4] are capable of generating high-quality images that fit natural language descriptions, and work has even improved in generating 3D meshes [5, 6, 7] and videos [1] from text prompt inputs. We call these systems Generative AI. While powerful on computer interfaces, Generative AI systems cannot directly make things in the real world.

Definition 1.1.2 (Low-level goal)**.** A low-level goal is a near-exact specification of a completion state for a given task. There is very little room for interpretation in whether the complete state of a task is similar to the low-level goal.

Definition 1.1.3 (High-level goal)**.** A high-level goal is an abstract specification of the completion state for a given task (1) that leaves room for multiple interpretations on how to complete the task and (2) there are multiple instances of the completion state that satisfy the given goal.

Since there are multiple interpretations of high-level goals, generative robots fill in gaps in the specified goal when making things. Beyond being able to comprehend these high-level goals, generative robots must also use other high-level reasoning about the things that they are making. For example, a robot must understand the cultural context of what it is creating so as not to create something offensive. Additionally the robot needs to have understandings of other high-level concepts like aesthetics or ethics.

1.1.3 Supportive, Human-Robot Co-Creativity

Creativity is something that people enjoy and generally do not want automated [2], and arguably computers cannot be creative at all [8]. It is therefore important that robots are designed to *support* human creativity, rather than replace or automate it. We theorize that robots can support creativity through at least three mechanisms: acting as a collaborator, inspiring new ideas, and motivating a person to create. We show that a robot can teach itself to collaborate with painting through self-supervised learning.

[1] https://openai.com/index/sora/

1.2 Creativity and Art

This thesis presents fundamental scientific contributions of robotics that are developed and evaluated in artistic and creative domains. In this section, we briefly comment on art and creativity to align readers with our definitions and positions on these matters.

Creativity

Margaret Boden defined a creative idea as being novel, surprising, and valuable. We simplify this definition to just novelty and value. Novelty means that the idea is new to a single person or to all people. Value is subjective and can mean many things to different people. For example, some people may value the provocativeness of an idea, and another may value the beauty of it.

AI is trained on data which often represent valuable, past solutions, so, models are able to predict solutions that fall into the distribution of previous valuable solutions. For example, a person values images of calico cats, creates a dataset of these images, and then trains an AI model to generate more pictures of calico cats. The model will be very good at producing these valuable solutions with sufficient training data. However, in terms of creativity, the model will likely lack novelty. The generated solutions will be derivative of the past training data. For example, if the calico cat image generator is fit well, it will never produce an image of a different cat that could surprise and wow the viewer.

This rhetorical argument shows that data-driven AI will lack the novelty needed to be creative. Furthermore, the more strong the model is fit to the data, the less likely that surprising novel samples will be created. Therefore, in this thesis, **we pursue AI as a support tool for human creativity**.

Art

It is perhaps impossible to properly define art, and any prior attempts to nail it down have been met with challenges by the finest artistic minds. For example, Marcel Duchamp's *Fountain* (1917) challenged the prominent ideas that art must have fine craftsmanship or beauty. This notion was again challenged with Maurizio Cattelan's *Comedian* in 2019 in response to an elitist art market.

Properly defining or debating what is and is not art is rarely a productive conversation. Deciding what can be considered "art" is either too broad a question or serves as gate-keeping to only allow a select few to engage in art as a practice. Rather than asking "is it art?", we encourage readers to consider questions that are deeper and can result in interesting debate, such as "Is the story the piece is telling compelling or relatable?", "What do you think about the colors used in this piece", or "Have you ever seen something like this" [9].

A large part of existing debates about what can be considered art is the distinguishing of art versus craftsmanship. Aaron Hertzmann points out the paradox of common opinions that craftsmanship may only be considered art once it reaches a certain level of technical expertise, but it cannot be considered art if it is purely technical skill and lacks emotion, intent, or other aspects that people associate with art [10].

This thesis makes fundamental robotics scientific contributions and is not intended to be an art project. However, **we hope that the contributions in the open-source systems presented here could be used to support someone expressing themselves by being inspired to make more artwork using this technology.**

1.3 Learning-Based Robot Intelligence

To tackle Generative Robotics, the domain of this thesis, we propose to use learning-enabled robot intelligence. There is a long history of robot planning approaches, including search, rule-based systems, expert systems, and control theoretical approaches. However, recent approaches tend to plan by modeling data, piggybacking on recent developments in neural networks and the capability to predict complex patterns from huge datasets or exploration in simulation. These approaches can learn complex patterns from the data. Since creative acts are so complex, we choose to use learning-based robot intelligence.

There are two primary methods for learning-based robot intelligence: learning from demonstration and reinforcement learning. Learning from demonstration, also known as behavior cloning, is a supervised learning problem in which a neural network models a set of demonstrations of a robot performing a task. These demonstrations are often provided through teleoperation or kinesthetic teaching. This process can be very tedious as it may require thousands of demonstrations to learn a task such that it can be reproduced under out-of-domain circumstances. Furthermore, if the robot embodiment, tools, or materials change, the demonstrations may need to be completely recollected.

In reinforcement learning (RL), a robot learns a policy by exploring an environment with a given reward function. The robot tries different things and determines how to get take actions to gain reward over periods of time. This can result in discovering interesting strategies, but it may require too much compute time for a system to discover these rare strategies. Generally, reinforcement learning agents are learning in a simulated environment, so these agents struggle to generalize to the real world as many things like deformable materials or liquids are challenging to simulate accurately. A final downside to using reinforcement learning is that it is challenging to write reward functions such that the agent will be able to learn efficiently. Often, reward is given only after taking many actions, and because the reward is so sparse in the explorations of the robot, it may not converge on a good policy. This is exacerbated when reward functions are very complex, such as the ones that would capture the

high-level aspects of Generative Robotics.

Model predictive control (MPC) involves optimizing a set of actions given a current state, dynamics model, and objective function. Since MPC simply finds a single solution at a time rather than RL which learns a whole policy, it generally can converge with more complex goals and dynamics. The downsides to using MPC for Generative Robotics are that it requires an accurate dynamics model, complex objective functions must be designed, and the goals are not complex or supportive out of the box. In this thesis, we introduce three self-supervised learning techniques to enable MPC to work for Generative Robotics.

1.4 Thesis Statement

In this thesis, we explore our proposed field of Generative Robotics (Sec. 1.1) using learning-based robotic intelligence approaches (Sec. 1.3). The primary limitations of using learning-based robotic intelligence for Generative Robotics are data related. Learning from demonstration requires too many samples to learn complex tasks and does not generalize to new tools, materials or actions. If text-to-image synthesis models like Stable Diffusion [4] required hundreds of millions of text-image pairs to train end-to-end, wouldn't a robot painter require as many text-painting demonstrations? Reinforcement learning also struggles to learn complex tasks and reward functions. In this thesis, we propose an approach to adapt model-predictive control to perform Generative Robotics tasks without excessive human demonstrations or infeasible numbers of simulations. Instead, our approach self-generates data to learn about its abilities and constraints in a process called self-supervised learning. We test our thesis statement in two Generative Robotics domains: Painting and sculpting. We demonstrate that our approach is able to collaboratively create paintings with human users and sculpt various deformable materials while planning in a visually aligned representation.

Thesis Statement

Self-supervised learning can enable model-based planning to understand real-world constraints, adapt to high-level goals, and support human-robot collaboration in Generative Robotics tasks.

1.5 Intellectual Merit and Contributions

This thesis introduces a generalizable self-supervision framework which contributes to the robotics research community at large. These techniques successfully enable model predictive control to perform Generative Robotics tasks.

Real2Sim2Real Dynamics Model We introduce a technique for self-generating actions to model the dynamics of complex materials using deep neural networks in a sample efficient way.

Planning in Semantic Representations We demonstrate a method for planning in semantic representations which can increase alignment between a person's goal for a robot and the robot execution.

Collaborative Goal Creation Generative Robotics have complex high-level goals, such as adding a tree to the background of a painting. We introduce a self-supervised learning technique to ground pretrained foundation models with the capabilities of the robot and ensure that generated goals are aligned with collaboration.

We first demonstrate these techniques in the robot painting domain, then prove that they can generalize to different action representations, state representations, tools, and materials with robotic sculpting.

2 Background

2.1 Related Work: Generative Robotics

This thesis attempts to formulate and solve a novel problem, Generative Robotics, which is distinct from Generative AI and Real-World Robotics problems in that it creates things from high-level goal inputs *and* in the real world, Figure 2.1.

2.1.1 Real-World Robotics

We consider systems as Real-World Robotics but not Generative Robotics if they do not have high-level goal inputs. Since low-level goal inputs are concrete and almost fully specified, it is more straight forward to develop an approach to making them. When systems can be highly engineered for a task, though, this can make things more straight forward as with printers and 3D printers. However, using more unpredictable

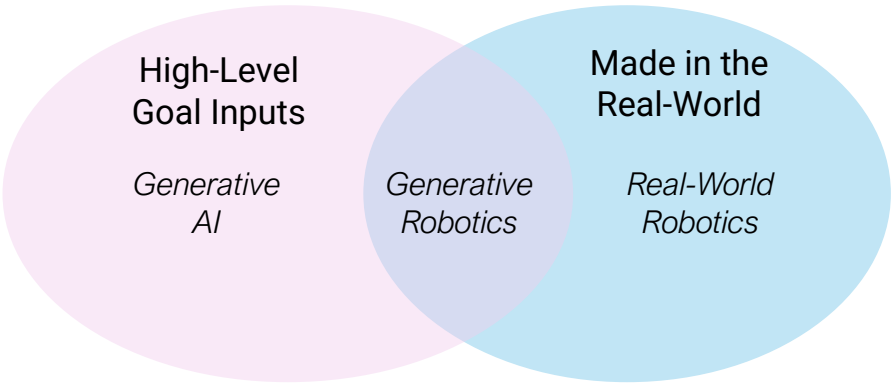

Figure 2.1: Generative Robotics merges the real-world capabilities of robotics with the powerful, high-level goal input capabilities of computer-based Generative AI.

materials and noisy tools creates a bigger challenge.

In the sculpture space, works like RoPotter [11] create clay pots using a pottery wheel and a rigid finger end-effector, RoboCraft [12] shapes clay into a discrete set of shapes with a robot gripper, and MonumentalLabs[1] is a company which can create stone sculptures by removing stone with a robot arm with a milling bit attachment. In 2D, many artists, such as Licia He [13], use XY plotting machines to create artworks with real world materials such as water colors or pens. GTGraffiti [14] uses real spray paint to create large artworks with a cable-driven parallel robot. Robot painting systems [15, 16, 17, 18] paint using real paint on canvas from image inputs. While these works are impressive due to their usage conventional tools and/or non-standardized materials, they do not have high-level goal inputs separating them from Generative Robotics: GTGraffiti and RoPotter create only the artwork demonstrated by a human, many robot painting works only work from image inputs [15, 16, 17, 18], and MonumentalLabs and plotter art systems create work using 3D modeling software or a complex network of vectorized curves. Beyond visual art, robots have been used for tending gardens [19, 20], sports arts like soccer [21], and dancing [22, 23].

2.1.2 Generative AI

While Real-World Robotics require highly-specified inputs, on computer-screen interfaces, there has been a recent boom in works that generate from more abstract inputs. We call systems that create artifacts on computer interfaces from high-level goal inputs, Generative AI. These include systems for text-to-image or text-to-3D where human users can give natural language descriptions of what they would like the system to generate. Recent developments in large-scale internet datasets, such as LAION [24] and Objaverse [25], and algorithms, such as Diffusion Models [4], have enabled generation from very flexible inputs. However, these systems do not generate real-world artifacts.

2.1.3 Generative Robotics

The most simple forms of Generative Robotics systems simply connect Generative AI and Real-World Robotics systems out of the box together. As an example, "Edmond de Balamy" is a famous AI generated image that was printed and sold for hundreds of thousands of dollars [26]. More complex systems can use more interesting materials and tools than an inkjet printer, such as Karimov et al. 2023 [27] who created reproductions of images generated from text inputs using Midjourney's [2] AI tools with a robot that paints. More intelligent systems integrate the challenges of the real-world into the planning of what the robot creates. DreamPainter [28] is a system that takes human speech input and produces marker drawings. The system plans to match

[1] https://www.monumentallabs.co/ourwork
[2] https://www.midjourney.com/home

the text from the speech to the marker strokes in a simulation, fully integrating the input to the action planning. This system worked well because there was an existing simulation that worked similarly to the robots abilities to use markers, however, would not generalize to other materials. Thus forms the holy grail of Generative Robotics, a system which can satisfy the user's high-level input using any materials and tools where the real-world *influences* what and how the robot creates things.

Recent developments in Large-Language Models (LLMs) have enabled high-level planning for robotics. For general robot manipulation, many works use LLMs or train Vision-Language-Action models to manipulate objects using language prompts [29]. Towards making objects, Blox-net [30] uses an LLM to plan actions which are simulated in a physics simulator. The planner picks robust plans for the simulation and runs them on a real robotic system. Blox-net is capable of creating real-world sculptures of blocks in a variety of shapes specified through natural language.

2.2 Related Work: Robot Learning for Making Things

There is a rich history and diversity to planning algorithms for robotics from search to expert systems to vision-language-action models. This section serves to introduce broad learning-based robotic intelligence planners methodology to help the reader understand how existing methodology could or would fall short of working for Generative Robotics tasks, such as painting and sculpting. We will go over several major categories of robot learning paradigms: imitation learning, reinforcement learning, and model-predictive control.

2.2.1 Imitation Learning & Learning from Demonstration

Imitation learning, a type of learning from demonstration [31], is a robot learning paradigm where a model learns how to predict actions that match the decisions of an expert demonstration. The expert demonstrations may be specified in a few different ways. In RoPotter [11], a human teleoperates a robot with a hand held controller to provide demonstrations for pottery making. Virtual reality is another method for demonstrating that has been successfully used for manipulation tasks [32]. Kinesthetic teaching can be useful when a robot's end-effector differs greatly from a human hand [33].

Imitation learning does not require any explicit modeling of materials. Instead, the dynamics and tool interactions are all modeled implicitly through the expert demonstrations. This has led to complex materials being used, such as clay [11, 34, 35].

When the number of demonstrations approaches massive proportions, these models are often referred to as vision-language-action (VLA) models. With so many demonstrations, these models can achieve very general behavior with very high-level

inputs, such as natural language. VLAs [29, 36, 37] have very general capabilities, such as performing pick-and-place, because of their massive training datasets. But as task complexity increases and goals push outside of training scenarios, the VLAs performance drops significantly [38]. Painting is a complex task where artists are constantly trying to make *new*, out-of-distribution samples. Imitation learning as a framework for painting may not be able to capture the complexity and creativity of the task no matter how large the dataset size. Training text-to-image models, such as Stable Diffusion [39], required hundreds of millions of text-image pairs. How many text-painting-demonstrations would be required to training an imitation learning model for painting? This would be quite infeasible. Additionally, the robot would need new demonstrations when new paint brushes or artistic settings are changed. Instead, for Generative Robotics tasks, we push for a model or simulation that can be used to maximize objectives rather than learning purely from demonstrations.

2.2.2 Reinforcement Learning

Reinforcement learning (RL) is a paradigm in which an agent explores a simulation or model and tries to learn a policy to maximize a learnt value function. It possible that the RL agent is exploring the real world, however, this usually is too slow to learn complex policies or tasks. In simulation, RL agents can explore thousands of options simultaneously [40]. A simulation can be replaced with a dynamics model or world model for gradient accelerated learning which can help with complex tasks [41]. The agents learn to maximize future rewards which can be specified by a user or can be inferred using complex models such as video prediction models [42] or LLMs [43].

RL has been used for Generative Robotics tasks before, such as painting [44], however these policies often have trouble transferring into the real world [45]. This sim2real gap is a common issue when using RL. A policy becomes so finely tuned to the model or simulation that it is unable to perform well in the real world. This is particularly challenging with materials that are difficult to model such as paint or clay [12].

An additional challenge of RL is that learning policies for complex tasks requires a great effort to enable the model to learn [46]. In theory, if an RL policy has enough time to explore, it can learn anything. However, in practice, training can be unstable and stall. Especially with complex objectives over long horizons with sparse rewards, an RL agent may never learn a policy. Common techniques to remedy this are to scaffold simpler rewards to help the agent learn via a curriculum [47]. Still, a complex task like painting may never converge. Past work in RL for painting has used highly unrealistic simulators that do not transfer well into the real world [48]. Our prior work adapted this environment to be more realistic, however, the RL policy could barely learn without scaffolding [45]. The FRIDA dynamics model is even more complex and our experiments with training RL policies using it were unsuccessful due to a lack of convergence.

2.2.3 Model Predictive Control

Model predictive control (MPC) uses a dynamics model, just as many variants of RL do; however, rather than learning a whole policy, MPC generates one solution at a time. A set of actions are optimized through a dynamics model to achieve an objective at each time step to determine what actions to take next. The model can be specified through hand designed physics, but since many dynamics are complex and the real world has variables that are challenging to capture, models are often learned through data [49] potentially with deep neural networks for model flexibility [50]. While these approaches can learn powerful dynamics models, they can be prone to overfitting due to the differences in training data and number of parameters. In this thesis, we introduce techniques to avoid overfitting while allowing a model to learn with minimal amounts of data.

The goal state for MPC can be specified by a human user or baked into a cost function. Generally, the goal states are simple, for example, a position and orientation in space. In Generative Robotics, though, the goal states are highly complex in contrast. For example, in collaborative painting, a goal may be to add a tree to the background of an existing painting. MPC algorithms do not have these collaborative and complex goals by default. In this thesis, we introduce a self-supervision technique to create a goal generator model.

The objective or cost function compares the goal state and the predicted state from the dynamics model. Generally in MPC, this is a simple comparison using mean-squared error or similar distance metrics. However, this will fail when the robot cannot exactly achieve the goal state. In painting, a user may want the robot to draw from a goal color photograph using just a black sharpie. To make more complex comparisons, an objective function can be learnt in Inverse Model Predictive Control or Model Predictive Actor-Critic but this requires training data or a proxy goal. In this work, we propose to perform comparisons in the objective function in a more semantic representation of states. For example, in painting, we compare deep features extracted from the image states using pretrained neural networks. This allows the robot to draw from color photographs even if it just has a black Sharpie marker because it is drawing the semantic content, rather than the individual pixels.

3 Overview

3.1 Generalized Approach to Generative Robotics

We outline a generalized depiction of our approach to Generative Robotics in Figure 3.1. The inputs are the desires of the human user and the current state of material. A planner, we call the Supportive Goal Planner, produces a target state that matches the desires of the user and is achievable with the real-world constraints of the robot. This target state is akin to a preview of the final output.

To generate the actions to create this goal state in the real world, the robot uses the Low-Level Action Planner. In this planner, the effect of the robot's actions on the state can be simulated using a dynamics model. A set of actions are randomly initialized and then optimized using gradient descent to decrease the semantic difference between the predicted and goal states. The semantic difference is a comparison made between the predicted and goal states in a more semantically aligned representation.

3.2 FRIDA Overview

In this document, we first show how our approach is applied to collaborative robot painting. We call this system FRIDA, A Framework and Robotics Initiative for Developing Arts. We depict the approach in Figure 3.2. The states for this implementation are RGB images. This is in comparison to our sculpting approach which represents state in depth maps.

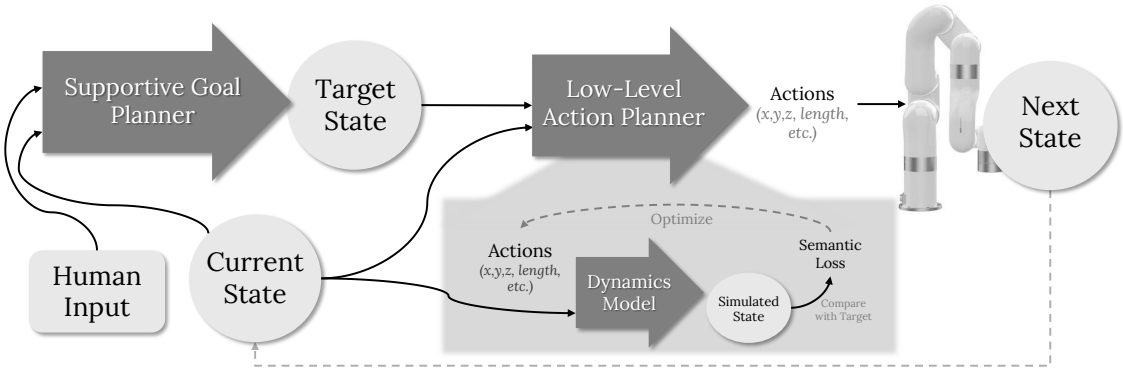

Figure 3.1: Generalized Approach Overview. In our system, input by a human is given along with the current state. The Supportive Goal Planner (Part III) creates a goal state that collaboratively uses the human input make changes to the current state. This goal is designed to be supportive to the user and achievable for the robot. The Low-Level Action Planner (Part II) iteratively optimizes a randomly initialized set of actions such that the dynamics model prediction of the effect of the actions matches the goal state in a semantic representation. After optimizing the actions according to the semantic objective, the actions are executed by the robot to produce the next real state.

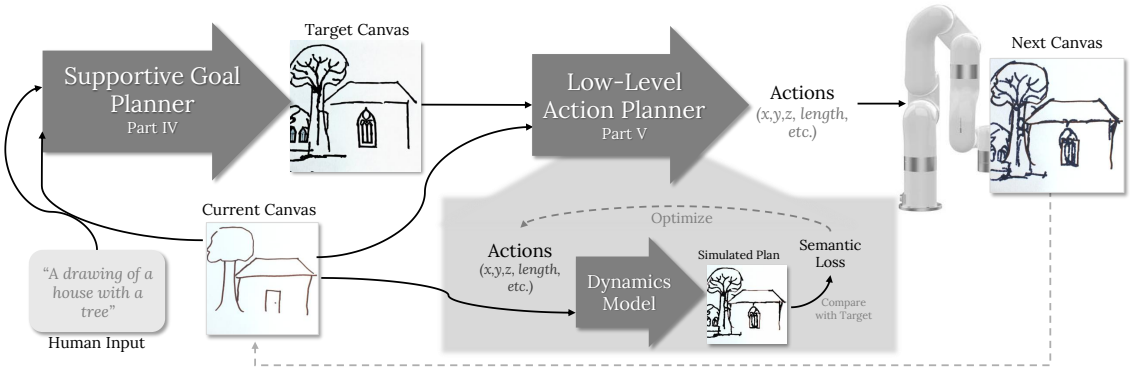

Figure 3.2: FRIDA Overview. FRIDA is a system for 2D Generative Robotics which can paint given human input. The Supportive Goal Planner produces an image of what the robot should paint, and the Low-Level Action Planner uses a dynamics model and semantic loss to optimize a set of actions to determine how the robot should paint it.

II

Low-Level Action Planner: *How* the Robot Paints

4 Dynamics Model and Semantic Planning for Robot Painting

4.1 Introduction

In this chapter, we introduce our self-supervised learning techniques for dynamics modeling and semantic planning for robot painting. These are used in the FRIDA robotic painting frameworks Low-Level Planner to plan actions to recreate the content of given images using paint brushes and paint. The goal of the Low-Level Planner is to take an image input and determine what actions the robot needs to take to accurately reproduce it given the current canvas state and the tools and materials it has available (FRIDA's embodiment can be seen in Figure 4.2). Reproducing an image with paint is challenging because of limitations in the tools, materials, and robotic actions may not be able to accurately represent the given image. Besides, it's challenging to properly define what it means to "accurately reproduce" a given image. For example, a pencil sketch of a color photograph may be considered an abstract, but accurate reproduction even though the low-level features of the two images are incredibly different.

Robot planning is often performed in a simulator. In order to reproduce an image, the transferring of a painting plan generated within a simulator into the real world must have a very small difference, or Sim2Real gap. This is particularly challenging in painting, as brushes and liquid paint behave very unpredictably.

So we need a simulator that can (1) have a small Sim2Real gap and (2) be used to plan abstractly such that a painting plan can be produced even if the tools and materials are unable to reproduce an input image with pixel-perfect accuracy. To solve this, we introduce a Painting Dynamics Model which uses real robot data to create a simulation environment, known as Real2Sim2Real methodology [54]. We ensure that the Painting Dynamics Model is differentiable, such that we can use neural networks within the planning algorithm to abstractly compare the current painting plan to the target image rather than comparing pixel-to-pixel.

Figure 4.1: FRIDA's Low-Level Planner is capable of planning of reproducing images. FRIDA is able to plan in simulation to reproduce an image even if it is not possible to be pixel-perfect due to limitations in the robot's tools. The Sim2Real gap is decreased by using real robot data to inform the simulator. Target image sources: [51, 52, 53].

4.2 Related Work

4.2.1 Simulated Painting

Stroke-Based Rendering (SBR) recreates a given target image using a set of primitive elements that usually resemble brush strokes of paint. Procedural SBR methods generally use rules and heuristics to generate the stroke plan [55, 56]. Planning-based SBR methods use search, optimization, or learning models such as Reinforcement Learning or Recurrent Neural Networks to generate a stroke plan with an objective of replicating an input image [44, 57].

Recent SBR methods expands the input space to incorporate high-level goals to generate brush stroke simulated paintings based on language descriptions and/or style specification [58, 59, 60]. While these methods present appreciable results in simulation, technical challenges specific to transitioning from simulation to real robots have not been addressed.

4.2.2 Robot Painting

There have been numerous robot-created paintings including notable works that had competed in an annual competition in 2016–2018 [17], but technical details of most works have not been published. Based on published works, existing robot painting

Figure 4.2: FRIDA's embodiments and workspaces - (Left) Ufactory Xarm Lite6 with a spring-loaded marker holder. (Center) Franka Emika FR3 with a small brush. (Right) Rethink Sawyer with a large brush.

approaches can roughly be categorized into two groups: engineered systems and learning-enabled systems.

Engineered robotic painting systems

Engineered robotic painting systems use well measured equipment to ensure that the planning environment is accurate to the real environment and use rules and heuristics for planning. The Dark Factory portraits [15] utilize a highly accurate robotic arm with known models of brush shape and size. They plan a full sequence of actions a priori such that the plan can be blindly executed. E-David [16] uses a simulated environment constructed to be similar to its painting equipment then draws strokes perpendicular to gradients in the target image. Harold Cohen's AARON system creates content to draw and paint based on stochasticity and rules he created and draws using a highly tuned engineered system. In general, the engineered systems are capable of high-fidelity reproductions of input images as they meticulously engineer to minimize the sim2real gap in their setup; however, they are not generalizable to different equipment or settings. Furthermore, these approaches generally do not support more than replicating a given image.

Learning-enabled robotic painting systems

Learning-enabled robotic painting systems generally use simulation environments to plan brush strokes and then execute the plan using a physical robot. Due to a huge sim2real gap, brush stroke plans based directly on simulation methods [58, 59] produce

poor-quality paintings or are even infeasible on real robot systems. It has been shown that additional constraints help reducing the sim2real gap to enable robots to paint according to a generated plan [28, 61], but such rigid constraints sometimes result in vague or imprecise outcomes. In line-drawing, [62] used reinforcement learning to learn both the SBR instructions and the low level robot instructions for the reproduction of sketches. Their approach is designed to plan once and execute a given plan as is without observation feedback in the loop. In painting, however, visual feedback is crucial as painting is a continuously evolving process [63].

4.2.3 Brush Stroke Modeling

Brush strokes can be represented using a height map and a color map as in [64] where Generative Adversarial Networks are used to map a user input trajectory into a synthesized brush stroke. In their work, both training and testing were done using data synthesized using a volumetric oil painting simulator based on WetBrush [65]. While the outputs appear impressive in simulation, the challenge still remains unanswered how such a simulated input can be translated into a real painting, for example, by a robot.

Wang et al. [66] use brush parameters such as the width, drag, and offset of the brush's bristles to create a very accurate brush stroke model. They use pseudospectral optimal control to optimize trajectories of brush strokes to fit the target calligraphy character, which works well with calligraphy where an initial path is given in a reasonably accurate form and the brush strokes are clearly separated by white space. In the painting domain, however, a more generalizable approach is needed due to the fact that the shapes of brush strokes used in painting are highly flexible and unconstrained and that brush strokes frequently overlap with previous ones.

4.3 Approach

4.3.1 Brush Stroke Action Parameters

Inspired by [66], we parameterize the space of brush strokes using three parameters as shown in Figure 4.3. In addition to brush shape attributes, i.e., the length l of the stroke, and the amount b that the stroke bends up or down, the thickness h of the stroke specifies how far the brush is pressed proportionally to the canvas. A brush stroke is parameterized by its shape, denoted by (h, l, b), the location coordinates on a canvas (x, y), orientation θ, and color ρ in the RGB format. The stroke trajectory can then be represented by a cubic Bézier curve where the horizontal coordinates are a linear interpolation between 0 and l, and the vertical coordinates are 0 at the end points and b in the center points.

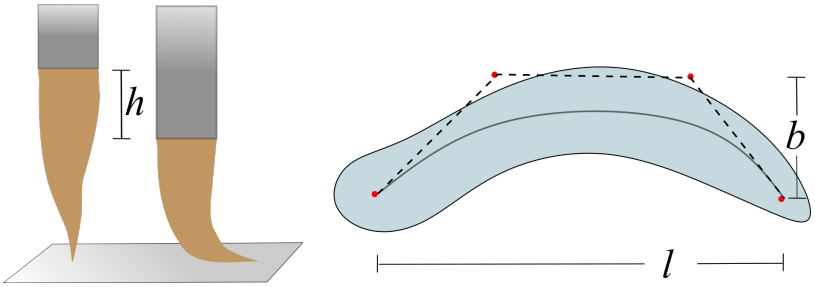

Figure 4.3: Our brush shape model has three parameters: thickness (h), bend (b), and length (l).

4.3.2 Real Data to Simulation

Whereas existing models such as [66] use only shape features of the rendered images that would require some model of a brush tool for a robot control interface, our definition of thickness connects the parameter space with a brush tool and a robot. During the calibration phase, we generate random brush strokes to train the `param2stroke` model, a Neural Network comprised of two linear layers followed by two convolutional layers and an bilinear upscaler, that translates a brush stroke shape tuple directly into the appearance map of the brush stroke. The brush stroke shape tuple can deterministically be translated into control inputs for a real robot.

A rudimentary approach to creating a differentiable, simulated robot painting environment would be to allow the robot to paint randomly and continuously to collect a large enough dataset of paired robot actions to the effects on the canvas to model this relationship. While this method works well in simulated environments [44, 60, 61, 64], where thousands of brush strokes can be produced on the order of seconds, generating a similarly large-sized real dataset is impractical. Painting in real life is slow, and if the brush or other materials were altered, the entire process would need to be restarted. Instead, we augment the dataset using existing differentiable functions, such as rigid transformations for positioning and orienting strokes and stamping methodology for rendering individual brush strokes onto an existing canvas, to allow our painting environment to be simulated with a reasonably small number of real brush strokes for modeling.

4.3.3 Differentiable Simulated Painting Environment

The stroke rendering process in our simulation is depicted in Figure 4.4: the `param2stroke` network translates the thickness, bend, and length parameters into a 2d magnitude map of the brush stroke's predicted appearance. This magnitude map is then padded such that it is the size of a full canvas. Then the map is translated and rotated to the specified orientation and location. The magnitude map is converted into an RGBA image, and then the stroke is applied to a given existing canvas. Strokes can

be layered upon each other to create a complete simulated painting. They can also be rendered onto a photograph of the real canvas for planning throughout the painting process. The whole rendering process is differentiable, meaning that the loss value computed using the rendered canvas can be differentiated, back-propagated through the simulator, and a Stochastic Gradient Descent algorithm updates the brush stroke parameters such that the parameters minimize the loss function.

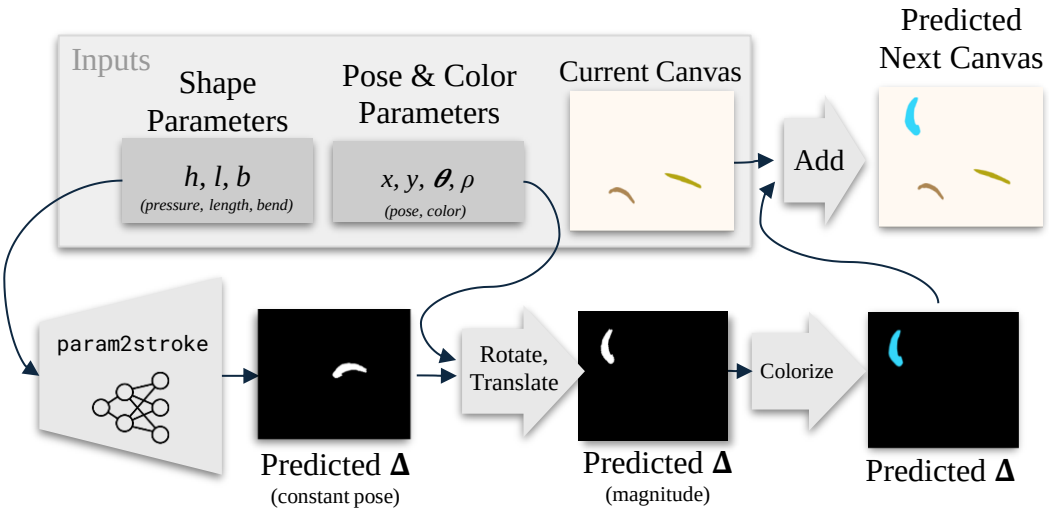

Figure 4.4: Paint Dynamics Model: The process of rendering a stroke, given its parameters, onto an existing canvas in our differentiable simulated painting environment.

4.3.4 Objective Functions

The enable the robot to create paintings from images even when it cannot reproduce it with pixel-level accuracy due to its tool and material limitations, we design objective functions to have semantic guidance, rather than pixel-level guidance. To test the differentiability and other capabilities of the renderer, we also employ a variety of objective functions from recent image synthesis literature. Each objective function has a loss function that compares the plan (p), which consists of a list of brush strokes parameterized by values in Sec. 4.3.1, to the target input (t) which may be language or an image. A plan p_{next} for the next time step is rendered into a raster image using a differentiable simulated environment (r). These objective functions can be used in different combinations to achieve high-level, artistic tasks, e.g., painting from language description with or without a specified style, painting images conceptually, or painting from a sketch.

$$l_1 = l_{text} = \cos(CLIP_{img}(r(p)), CLIP_{text}(t)) \tag{4.1}$$
$$l_2 = l_{style} = EMD(VGG(r(p)) - VGG(t)) \tag{4.2}$$
$$l_3 = l_{print} = ||r(p) - t||_2^2 \tag{4.3}$$
$$l_4 = l_{semantic} = ||CLIP_{conv}(r(p)) - CLIP_{conv}(t)||_2^2 \tag{4.4}$$
$$p_{next} = \min_p \sum_{i=1}^{4} (w_i l_i), w_i \in \{0, 1\} \tag{4.5}$$

Image-Text Similarity Objective

(Eq. 4.1) This objective optimizes the brush stroke plan (p) such that the cosine distance between the CLIP [67] embeddings of both the painting and the language description (t) is minimized, guiding the painting to resemble the content of the text, as is common in recent CLIP-guided text-to-image synthesis methods [58, 59, 68, 69].

Style Objective

(Eq. 4.2) Given an example style image, the style objective guides the painting to resemble the colors, shapes, textures, and other style features of the given image. This objective was created for style transfer methodology [70, 71]. The style objective minimizes the Earth Mover's Distance (EMD) between style features that are extracted using a pretrained object detection model (VGG [72]), from the brush stroke plan (p) and the style image (t).

Simple Replication Objective

(Eq. 4.3) Image replication is not considered a high-level goal. Instead, it is a straightforward minimization of the L_2 distance between the rendered brush stroke plan and the target image (t).

Semantic Replication Objective

(Eq. 4.4) Following [73], features can be extracted from the convolutional layers of CLIP which are rich in both semantic and geometric information. For a high-level semantic replication objective, we minimize the L_2 difference of features extracted from the target image and painting from the last convolutional layer of CLIP ($CLIP_{conv}$).

4.3.5 Planning Algorithm

We depict the painting planning algorithm in Figure 4.5. At the start of the painting process, a user gives inputs and decides which loss functions to use. For

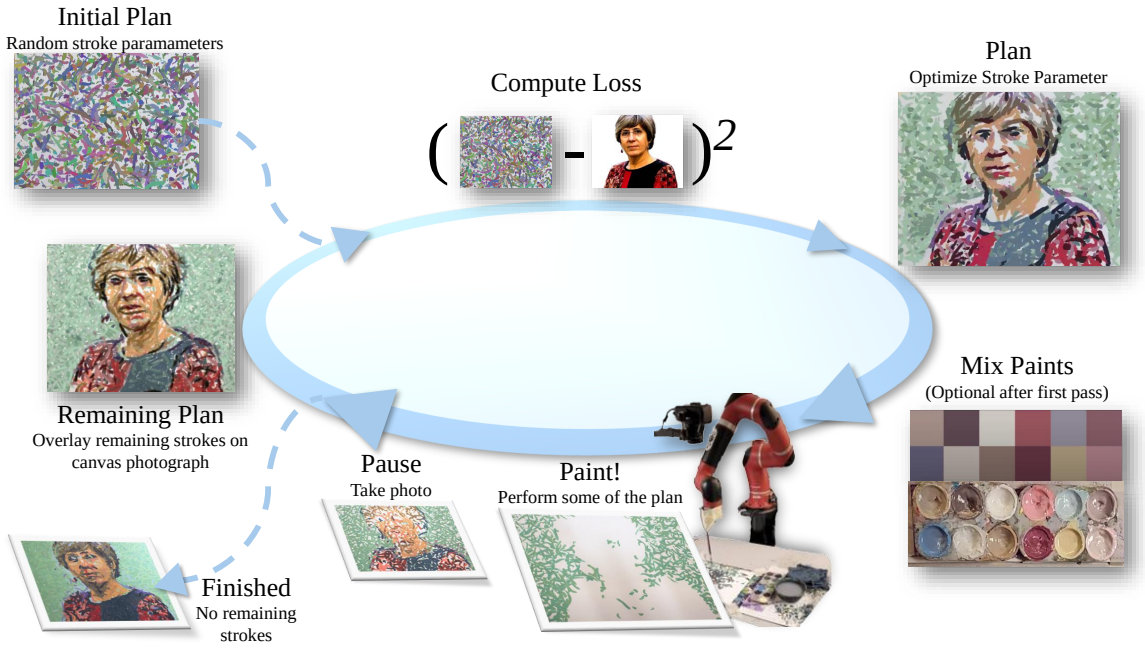

Figure 4.5: Painting Execution Algorithm - An initial plan is made by randomly initializing brush strokes. A loss value is chosen (l_{print} is displayed), then gradient descent is used to optimize the brush stroke parameters to decrease the loss. After optimization, the colors of the painting are clustered and displayed for the user to mix. The robot paints a set amount of brush strokes. The robot takes a picture of the partially complete painting, overlays the remaining strokes, then optimizes the strokes once again to account for prior noise and error in execution. This process repeats until all strokes are executed.

example, they may give a style image and use l_{style} and a language description with l_{text}. They also decide how many brush strokes to use in the painting. Our painting action space is comprised of discrete brush strokes, each parameterized by values described in Sec. 4.3.1. A plan is made up of an ordered list of brush strokes. Brush strokes are randomly initialized by sampling uniformly over the brush stroke parameters.

The goal of our painting algorithm is to find a plan that minimizes the weighted sum of the user specified loss functions, Eq. 4.5. Because the rendering pipeline is differentiable, we can compute the derivative of the loss values with respect to each of the brush stroke parameter's values. We use Adam, a variant of Gradient Descent, to update the brush stroke parameters given this derivative to decrease the loss values. The actions are optimized for a specified number of iterations. After 50% of the iterations are complete, the color parameters are discretized to 12 colors using k-means clustering (Figure 4.6). We only performed this after 50% of the optimization so that the colors can naturally find optimal values before being binned. The strokes are also sorted from lightest color to darkest color to avoid bleeding of dark colors into areas of the painting that should be light. After the set number of optimization iterations, the

Figure 4.6: Discretizing Colors. Left – Our robots plans with arbitrary colors for each stroke. Right – After using k-means clustering to discretize the colors to just 12.

12 colors must be mixed by hand by referencing a rendering of them on a computer screen and provided to the robot.

The robot begins to execute brush strokes from the initial plan. After painting a set number of brush strokes (we use 20 in practice), the robot takes a photo of the canvas and then updates the remaining brush strokes in the plan. The update process uses the same objective function as before, but now only the remaining brush strokes are optimized and they are rendered onto the photo of the current canvas. This process is repeated until all brush strokes from the plan are executed.

4.4 Robot Setup Details

We used a Rethink Sawyer robot [74] as a machine to test our approach. Any Robotics Operating System (ROS) compatible machine with a similar morphology to the Sawyer could feasibly be adapted to execute our approach with only minimal changes to the robot interface code.

A photograph of our painting equipment and setup can be seen in Figure 4.2. We use a Canon EOS Rebel T7 to perceive the canvas. For all examples in this paper, we used 11×14 inch canvas board as painting surfaces. Premixed acrylic paints are provided to the robot in palette trays with up to 12 color options available. Alternatively, from an initial painting in simulation, the colors are discretized to a user-specified number using K-Means cluster; palette preparation is performed accordingly by a human. A rag and water are provided for the robot to clean paint off of the brush, which is performed when switching colors. The brush is rigidly attached to the robot's end effector and is always held perpendicular to the canvas. Indirect, diffused lighting is necessary, since direct lighting can cause too much glare from the wet paint into the camera. The locations of all the painting materials (canvas, paint, water rag) with respect to the robot are explicitly programmed. We use a machine with an NVIDIA Quadro GPU that has 8Gb of CUDA memory for planning. A

painting with 1000 strokes takes roughly 3 hours to complete comprised of 30 minutes of calibration, 15 minutes for the initial painting plan, 15 minutes for paint mixing, and 2 hours of the robot actually painting.

4.5 Results

Figure 4.7: Depictions of interpolating between minimum and maximum values of each of the three stroke shape parameters with the trained `param2stroke` model.

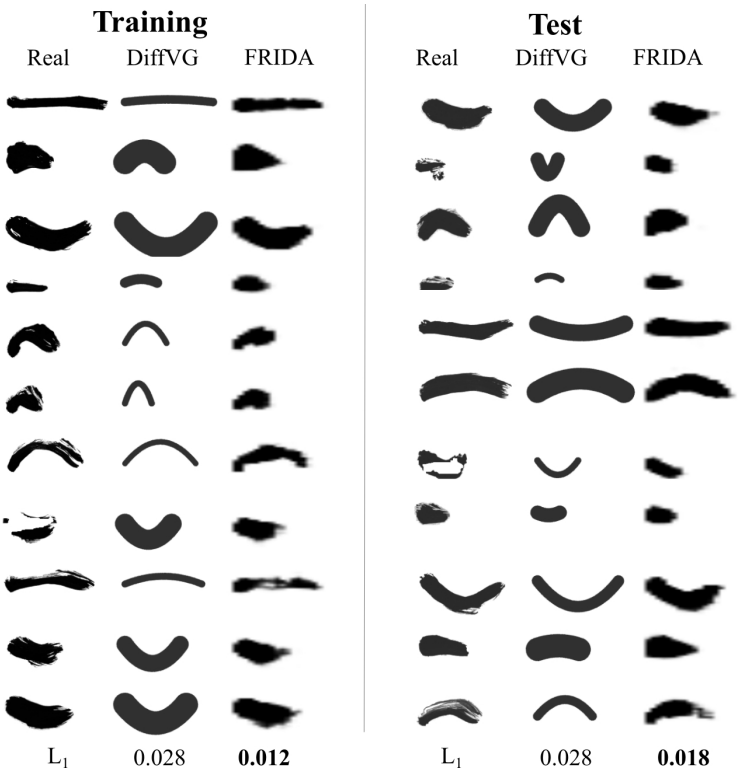

Figure 4.8: We compare using DiffVG [75] and FRIDA's `param2stroke` model for modeling brush stroke shapes. The average L_1 distance computed on 50 samples between the modeled and real brush strokes is displayed at the bottom.

4.5.1 Simulated Painting Environment

The trained `param2stroke` model produced strokes with continuous parameter values as seen in Figure4.7. Figure 4.8 shows the difference between real brush strokes and FRIDA's modeled brush strokes using the same input parameters. We also compared these strokes to DiffVG [75] which was used for brush stroke planning in [28]. The average L_1 loss between the modeled and real strokes was significantly (p-value $< .01$) less for FRIDA's stroke model than DiffVG.

We qualitatively compared our approach to Huang et al. 2019 [44] and Schaldenbrand & Oh 2021 [61]. Figure 4.9 compares the brush strokes in early stages of painting simulation where we can observe drastic differences.

Figure 4.10 shows the comparison in terms of the sim2real gap for entire paintings. In simulation, Huang et al. 2019's Reinforcement Learning (RL) model was able to almost perfectly replicate a given image due to their unconstrained stroke model, e.g., allowing strokes that are huge in size and have varying opacity; however, when we fed the strokes to a painting robot, the produced painting was vastly dissimilar to both the simulation and target image. Schaldenbrand & Oh 2021 constrained the brush stroke parameters (length, width, color, and opacity) such that a robot was more capable of executing the strokes; however, the constraints made it challenging for their RL model to accurately replicate a target image. For the proposed approach, we used our simulation to recreate the target image using the Simple Replication Objective (Eq. 4.3) and did not re-plan with perception for fair comparison. Our proposed approach showed clearly visible improvement in recreation both in simulation and real painting.

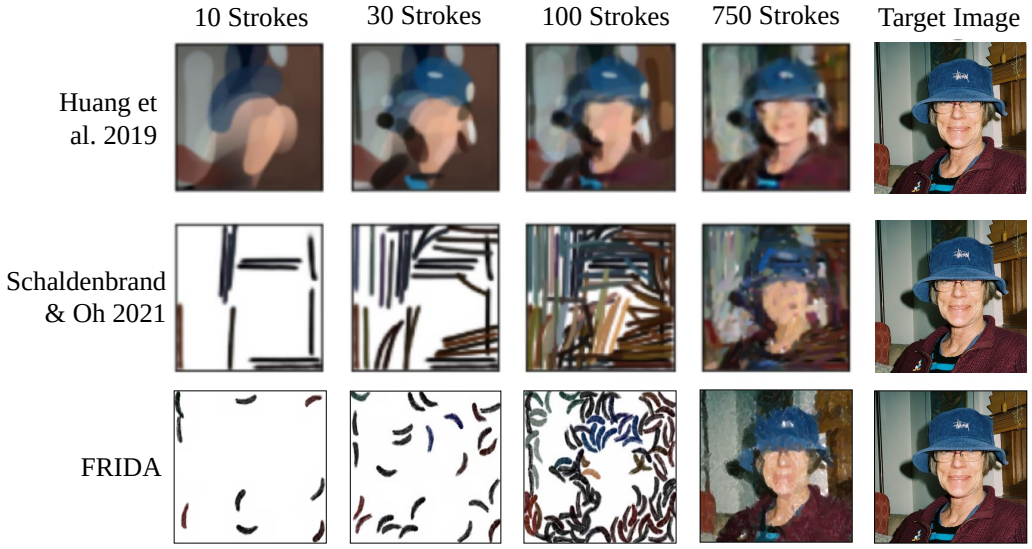

Figure 4.9: Stroke Shape Feasibility - Comparing the simulation environments of three painting methods painting with various numbers of brush strokes. Top to bottom: Huang et al. 2019 [44], Schaldenbrand & Oh 2021 [61], and the FRIDA Low-Level Planner.

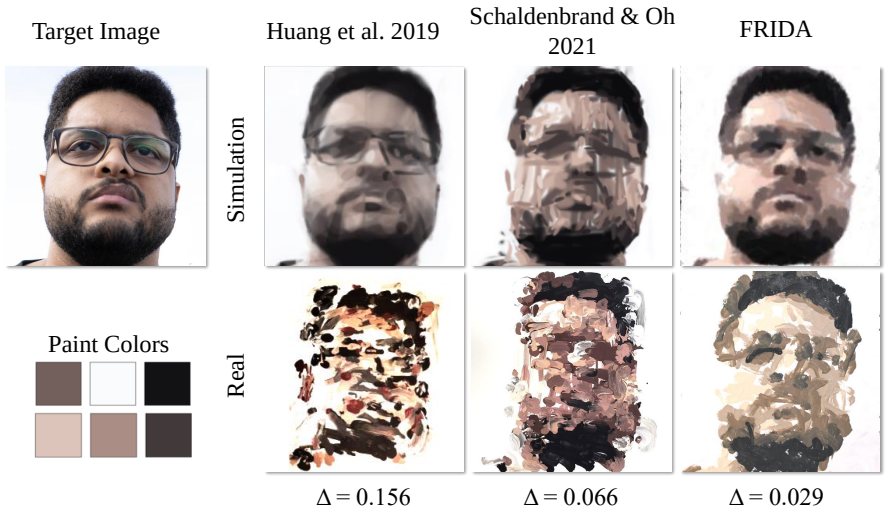

Figure 4.10: Sim2Real Gap - We compare the Sim2Real gap between FRIDA and two existing methods. The MSE between the simulated plan and the real painting is displayed below each pair.

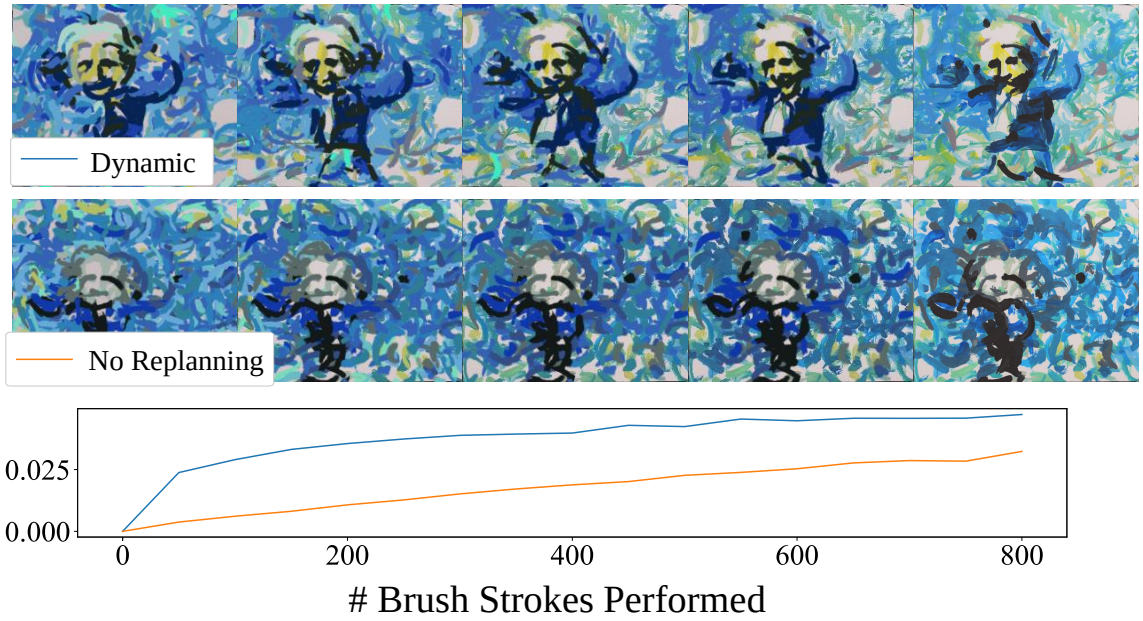

Figure 4.11: FRIDA painting with text input "Albert Einstein Dancing" in the style of van Gogh's *The Starry Night* with and without replanning. The left most images are the initial plan followed by the plan after 200 brush strokes performed until the last column which is completely real paint. Below, the mean squared error between the current plan versus the initial plan is plotted.

Figure 4.12: Painting from Style and Text Inputs with the Low-Level Planner - FRIDA's Low-Level Planner is capable of planning from text and style inputs seen in combinations above by optimizing brush stroke parameters to create a painting that matches CLIP and Style features of the inputs.

4.5.2 Dynamic Planning and Adaptation

We painted with and without FRIDA's dynamic replanning system and plotted the deviation from the initial plan in Figure 4.11. Without replanning, the difference between the current and initial plan grew linearly as the plan is executed from simulation to reality stroke by stroke. With replanning, the plan changed more significantly from the initial plan as the algorithm adapted to the stochastic execution of the plan, resembling the creative process of human artists [63].

4.5.3 Planning in a Semantic Representation

Painting from Language Description with Specified Style

We painted from language descriptions and given examples tyle images by concertedly optimizing the Style Objective (Eq. 4.2) and the Image-Text Similarity Objective (Eq. 4.1). Results can be seen in Figure 4.12.

To retain the style image's composition, we do an initial optimization to replicate the style image. The initial brush stroke plan is now in a local minimum which will be adapted with the full style and text objectives. Figure 4.12 shows that faces and

colors appear where they were initially located in the content image thereby providing a method of transferring compositional elements of style.

Painting Images Conceptually

We compare painting using the Simple and Semantic Replication Objectives in Figure 4.13. We hypothesized that the Semantic Replication Objective would better capture high-level content of the target image. To test this quantitatively, we recruited 103 Amazon Mechanical Turk participants to (1) "select the painting that looks the most like the target image" and (2) "select the painting that captures the high-level ideas of the reference image's scene better" and to explain how they made their decision. We refer to these surveys as the replication and high-level questions, respectively, and they were conducted separately. Simulated paintings were used to avoid noise generated by human error in palette preparation of which six pairs were generated with 10 evaluators for each question, painting pair. 73.3% and 68.3% of participants selected Semantic Replication Objective paintings for the replication and high-level questions, respectively. These two averages were both significantly larger than 50% at a p-value of 0.01 and were not statistically distinct using a t-test. While the two questions were different, we noticed that participants claimed to use many of the same features to make their decisions for each question which included colors, shapes, and particular details such as grass and clouds. A breakdown of selections for each painting pair is in Figure 4.14.

Figure 4.13: FRIDA's paintings using the Simple Replication Objective (Eq. 4.3) versus the High-Level Semantic Replication Objective (Eq. 4.4).

Replication	**70%**	**100%**	**100%**	**90%**	50%	30%
High-Level	50%	**80%**	**100%**	**70%**	**80%**	30%

Figure 4.14: Reproduction Ability Study - Results from two surveys assessing how well paintings replicated the target image and how well they retained the high-level content. Percentages shown are preferences for Semantic Loss (Eq. 4.4) paintings. Bolded numbers show when the Semantic Loss was more commonly chosen over the Simple Loss. In most examples, the Semantic Loss produced paintings that more closely resembled the target image in both exact replication and high-level reproduction.

Sketching

What if the robot only has a black marker? We adapt the Painting Dynamics Model to only use black as a color and not optimize over the color parameter. In Figure 4.1, an example of a drawing from a photograph can be seen. This was performed with the Semantic Replication Objective. It is especially important to use the Semantic Replication Objective because the painting (drawing in this case) will look so significantly different from the input photograph. We show the differences between using the Semantic and Simple Replication Objective with drawings in Figure 4.15. The Simple Replication Objective is unable to produce a drawing that captures the likeness of Frida Kahlo from the target image.

4.6 Limitations

Our dynamics model makes some simplifying assumptions that create some limitations. First, the dynamics model assumes that brush strokes are independent, which means that there is no modeling of wet painting mixing on the canvas. This could potentially be fixed by including an input of current state into the dynamics model, however, wet paint mixing is a complicated interaction. In our previous work, we

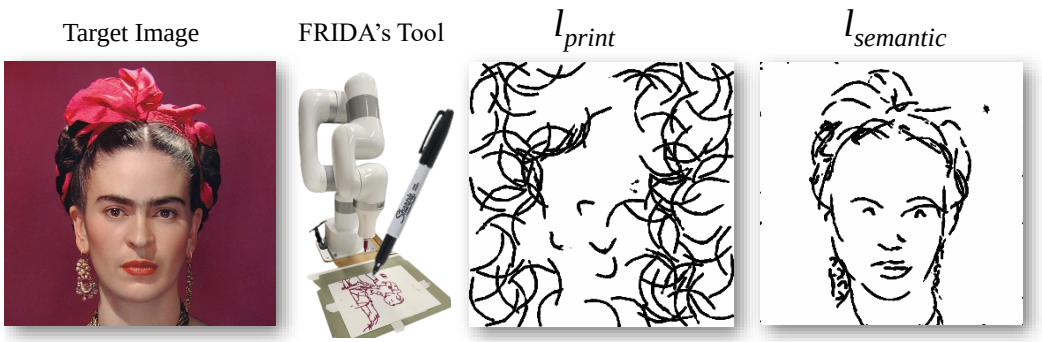

Figure 4.15: Drawing with Black Marker: With only a black marker and a limited number of strokes we compare using FRIDA's Low-Level Planner with the Simple (l_{print}) vs Semantic Replication Objective ($l_{semantic}$).

developed some techniques to model multi-colored paint mixing [76]; however, more work needs to be performed to streamline this into a dynamics model.

While our proposed approach greatly reduced it, there is still a Sim2Real gap in our dynamics model. Some reasons for this imperfection are the real stroke dataset being too small, noise and inconsistent control on the robot performing the brush stroke, and limitations in the neural network trained to predict the appearance. Additionally, in this chapter we only test with Bèzier curve action representations which are not very expressive compared to the stroke trajectories of human artists.

There are also limitations on our planning algorithm side. It is susceptible to being stuck in poor local minima of the loss functions. For instance, if the initialized brush stroke plan does not have many strokes in a highly-detailed region of the painting, then the algorithm likely will not be able to move the positions of the strokes to that region to properly capture all of the details.

Lastly, the planning algorithm is slow. Paintings, such as those seen in Figures 4.1 and 4.12, which contain a few hundred brush stroke actions, require roughly 15 minutes to optimize for 1000 iterations on an NVIDIA Quadro GPU.

4.7 Conclusions

In this section, we presented FRIDA's Low-Level Planner for robot painting which includes our dynamics modeling and semantic planning techniques.

4.7.1 Self-Supervised Learning for Brush Stroke Dynamics Modeling

When performing model-based planning in robotics, it is common to use an off-the-shelf simulator which uses hand designed physics formulations to model the dynamics

of actions and states. Off-the-shelf simulators often need to be grounded and altered to better fit a robotic system. In our experiments (reported in Figures 4.10 and 4.8), the off-the-shelf simulators had huge Sim2Real gaps. Our attempts to alter it by hand with rules decreased this gap, but it was still very inaccurate (Figure 4.10). Using real robot data to completely train a dynamics model resulted in a very small Sim2Real gap (Figure 4.10). Although we are not the first to use data to tune a simulator [49, 54] or to train a neural network for dynamics [12], the techniques we introduce allow the model to be accurate without reducing the dimensionality of the states and with fewer than 100 self-generated actions for training.

4.7.2 Planning in a Semantic Representation

In robotics, it is crucial that the goals of the robot align with the goals of the human user. In this chapter, we showed that when humans want to paint from an image, they want to paint the *content* of that image rather than making a pixel-perfect reproduction. We supported this by planning in a semantically aligned representation (visual features extracted using pretrained neural networks) rather than just the pixel-space. In our study (Figure 4.14), we observed that participants thought the images planned in a semantic representation looked more like the source materials than the pixel representation even though the pixel representation was more similar in mean squared error.

Planning in a visual latent space is technically challenging because the neural network that extracts the features (CLIP [67]) is deep and complicated. It is a challenge to optimize a set of robotic brush stroke actions to fit the semantic representation objective. We were able to achieve this using gradient-based optimization, which was possible because the dynamics model we created was fully differentiable. Otherwise, optimization would be infeasible with techniques, such as evolutionary strategies, random sampling, or training a policy with reinforcement learning.

5 Brush Stroke Diversity through Demonstration

> "The 'same' content represented in a different form—in a different medium or mode or style or language—is *not* the same: what is the same through all variations of the form is only a tenuous abstraction, a *précis* of the full content."
>
> —— Duncan Robertson, The Dichotomy of Form and Content [77]

In this chapter, we describe how learning from demonstration can be used to improve the action representation of the FRIDA robot painting system. FRIDA uses parameterized Bézier curves which are relatively unnatural and do not reflect the diversity of human actions in drawing and painting. In this chapter, we collect human-made brush strokes using motion capture. We modeled the trajectories with a

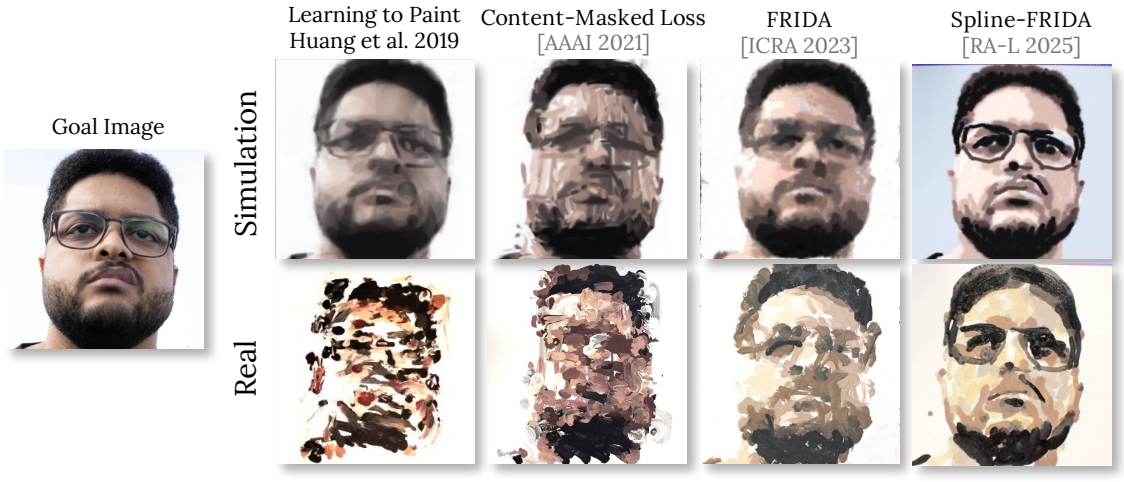

Figure 5.1: Spline-FRIDA's flexible brush stroke shapes allow for details like the person's glasses to be captured with a few defining strokes compared to the limited Bézier curves of the other methods.

variational auto-encoder, but also found that the existing FRIDA dynamics model cannot properly represent such diverse strokes. In this chapter, we introduce Spline-FRIDA which presents a novel brush stroke dynamics model which can model diverse, human-like strokes with only a few dozen, self-generated samples.

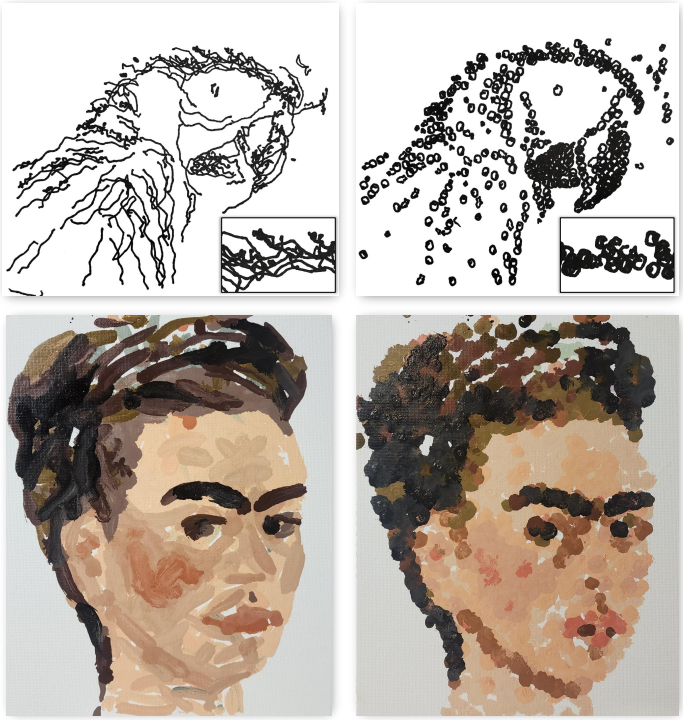

Figure 5.2: Spline-FRIDA drawings and paintings in different styles. These are two pairs of artworks of made by our system. The left paintings use longer, zig-zagging strokes, while the right ones are composed of small circles and dots. While each pair depicts the same content, the stroke style vastly changes the appearance and vibe of each work.

This work was led by Lawrence Chen during his Master of Computer Science degree.

5.1 Introduction

Paintings and drawings are used to convey messages of emotion, cultural values, and shared experiences. While these aspects can be conveyed by the objects or subjects within the painting, style is perhaps just as important to expressing those messages [59, 77]. In the visual art space, there is evidence that people find the style of an image to be even more crucial than its content when interpreting the meaning of a generated image [59]. In particular, patterns in the shapes of individual strokes within a painting can contribute to the overall style and aesthetic of an artwork. Some

examples can be seen in Figure 5.2, with the left drawings using long strokes and the right drawings using small, circular ones. In both cases, the stroke shapes are crucial for defining the painting's style and therefore the expression of the message that the artist intends to convey.

Furthermore, if robots are to support humans in the creation of artwork, it is important for the robot to have flexible styles of strokes for the user to specify either through choice or demonstration. Many artists do not wish to automate the artistic process [1, 78], but some are open to co-creative assistants [2, 79, 80, 81]. The work operates on the assumption that giving more creative control to a user co-creating with a robot over the style of the image, allows them to feel more ownership over the artwork that they create with the robot.

Prior work has mostly focused on planning paintings using basic stroke representations such as Bézier curves [44, 82] and only fixating on the style of the overall image [59, 60]. In this paper, we focus on how intra-stroke style control can be implemented.

Our work uses motion capture to record human drawings with real-world brushes and markers on paper. We model these recorded trajectories with an autoencoder, TrajVAE. We also introduce a novel brush stroke dynamics model, Traj2Stroke, which predicts the 2d outline of stroke given its trajectory.

5.2 Related Work

Stroke-Based Rendering (SBR) involves arranging primitive shapes to create an image, often with the goal of replicating some target image. Some recent works use forward prediction methods, in which a neural network learns to output the next stroke to add [45, 48, 83], while others use optimization-based methods, where stroke parameters are passed through a differentiable rendering pipeline and optimized via backpropagation [82, 84, 85].

5.2.1 Stroke Primitives

Most SBR research is focused on global planning and propose new algorithms to arrange stroke primitives. On the other hand, there has been little research into how the stroke primitives themselves should be defined. Some works use definitions that would be difficult to replicate on a physical robot. For instance, Learning to Paint defines strokes as translucent Bézier curves with arbitrary thicknesses [48]. Schaldenbrand et al. found that when their system was restricted to outputting more realistic brush strokes by making them opaque and limiting the sizes, the quality of generated images suffered [45]. Paint Transformer uses a mask of a brush stroke that can be transformed, resized, and recolored [83]. This arbitrary sizing of strokes without loss of precision would be very difficult to implement in hardware.

Based on human art, many drawing tools, such as markers or brushes, can inherently be versatile and adaptable enough to produce a wide range of stroke styles. Specifically, altering the paths of individual strokes can result in diverse styles. This has been observed and researched extensively in the context of human handwriting replication [86, 87], but only to a lesser extent for drawings. We hope to further explore how to define stroke primitives by explicitly modeling the style of stroke trajectories used in a drawing.

5.2.2 Differentiable Rendering

In SBR, differentiable renderers are modules that take in stroke parameters and output a rendered image. They differ from traditional renderers in that gradients of the image with respect to the parameters can be obtained. Having access to such a module is a crucial assumption of many modern SBR planners.

Learning to Paint [48] takes a reinforcement learning (RL) approach to SBR. Despite the fact that RL does not inherently require a differentiable environment, they found that using a differentiable renderer greatly boosted the system's performance and convergence rate compared to a model-free method. This is mainly because differentiable environment allows for end-to-end training of the RL agent. Paint Transformer [83] also makes use of differentiable rendering so that a loss can be backpropagated from the output image all the way back to its stroke predictor. These examples show that differentiable rendering can be useful even in methods that are not optimization-based.

DiffVG [88] is a popular library for differentiable 2D rasterization that has been used in many optimization-based SBR methods [89, 90, 91]. It supports rendering arbitrary parametric curves, either open or closed, including polygons, ellipses, and polylines. Due to its popularity, we also considered using DiffVG to model Sharpie marker strokes for this work. However, we discovered that out of the box, the DiffVG library does not support rendering polylines that are differentiable with respect to stroke thickness. DiffVG lines are only differentiable with respect to the control points. Furthermore, DiffVG decouples strokes into a boundary shape and a fill color, which we found to be too restrictive because it does not allow us to model the gradual dropoff in darkness from the center of a stroke to the outside. Thus, we choose to implement our own differentiable renderer, Traj2Stroke, which is specialized for rendering polylines.

5.3 Approach

5.3.1 Overview

Our approach to stroke modeling and rendering consists of (1) capturing and processing human demonstration data using motion capture technology, (2) modeling these trajectories by training an autoencoder, TrajVAE, (3) using Real2Sim2Real

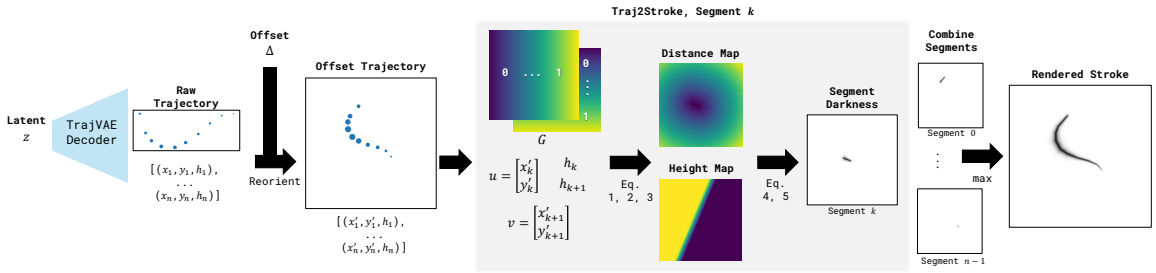

Figure 5.3: Traj2Stroke. The inputs are a latent vector z and an offset Δ. z is fed through the decoder of a TrajVAE, generating a raw trajectory, which is then rotated and translated according to Δ. We then process the trajectory segments independently, obtaining darkness values for each. Finally, we take the max darkness over all segments.

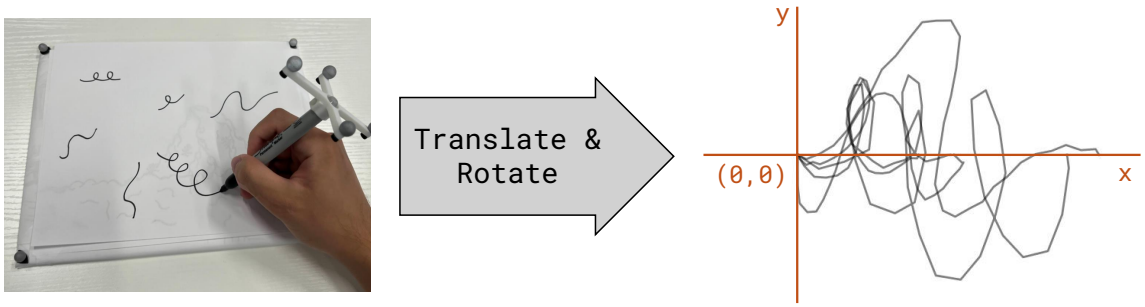

Figure 5.4: Mocap setup. We use a motion capture system to track the position of the canvas and pen over time as an artist draws. Three mocap markers are placed along the corners of the canvas, and four are mounted at the end of the pen. The trajectories of each stroke are extracted, then rotated and translated such that the start of the trajectory is (0, 0) and the end point is on the x-axis.

methodology to fine-tune our novel rendering approach, Traj2Stroke, and (4) planning using gradient descent to optimize a set of brush stroke parameters through our dynamics model to decrease the feature-space loss between a given image and the predicted painting.

5.3.2 Motion Capture Drawing Recording and Processing

We utilize a motion capture system consisting of OptiTrack cameras and Motive software to capture human brushstroke trajectories. While the artist sketches, we continuously track the positions and orientations of the canvas and pen. Using a manual measurement of the length of the pen, we are able to calculate the position of the pen tip and determine its distance from the canvas. If this value is below a threshold, we consider the pen to be in contact with the paper. Consecutive positions

where this is the case are merged into trajectories. Each trajectory is then standardized by translating it to the origin and rotating it to be horizontal (ending at $y = 0$), as seen in Figure 5.4, to reduce variation for sample-efficient modeling. It is worth noting that this normalization has a tradeoff: it assumes trajectory style is not affected by position/rotation on the canvas. We also resample each trajectory to have exactly n points (in practice $n = 32$).

Thus, each human brushstroke trajectory is modeled as a polyline (piecewise linear) going through n control points. This polyline is encoded as a $n \times 3$ tensor. The coordinates (x, y, h) of each control point are defined by x and y as horizontal displacements (in the plane of the canvas) and h as vertical displacement (elevation of the brush above the canvas).

5.3.3 TrajVAE

After collecting and processing the motion capture data, we train variational autoencoders [92] to model these stroke trajectories. We name these TrajVAEs. During training, a TrajVAE takes a trajectory as input, passes it through an MLP encoder that compresses it to a latent vector of size 64, and then sends it through a MLP decoder to turn it back into a trajectory. We minimize the mean squared error between the input and output trajectories.

We typically record between 20 and 200 human-drawn trajectories per drawing, but found that this is not enough data to robustly train a TrajVAE from scratch. Instead, we pretrain each TrajVAE on trajectories aggregated from multiple recording sessions, then fine-tune it on a single session to capture a more specific style. Each model converges very fast (less than a minute) and only requires a few (<20) trajectories in the fine-tuning dataset.

During the planning phase, only the VAE decoder is used. The design of the overall pipeline is modular so that different VAEs can be swapped in, allowing us to change the stroke style with no need for additional training.

Our motion capture device struggles to capture the vertical position of the drawing utensil's tip with enough precision. This is because a small height difference can drastically affect the thickness of a stroke. Thus, rather than explicitly modeling the height with TrajVAE, in practice we optimize it as separate stroke parameters during the planning process.

5.3.4 Traj2Stroke Model

The TrajVAE model outputs a trajectory that the robot should draw, but to predict the appearance of the stroke given this trajectory, we developed a novel rendering approach that we call Traj2Stroke. Traj2Stroke takes a trajectory, as well as positional and rotational offsets, and renders it as a grayscale image with height and width

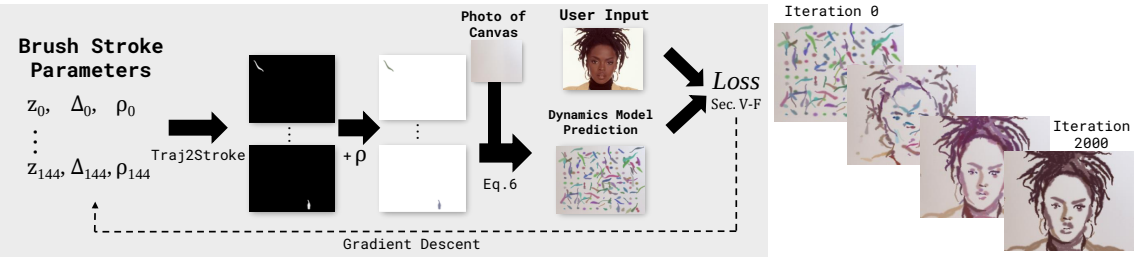

Figure 5.5: Planning a Painting. As described in Section 5.3.6, Spline-FRIDA plans a painting by optimizing the brush stroke parameters through the dynamics model to decrease a features space loss between a given image and the planned painting. Whereas FRIDA models brush strokes as simple Bézier curves, Spline-FRIDA uses trajectories which enable highly flexible brush strokes.

dimensions, $H \times W$. Importantly, this process is differentiable so that the planning process can backpropagate through it.

To train this model, we randomly sample trajectories from TrajVAE, execute them on the robot, and take before/after pictures of the canvas for each stroke. Next, we input the sampled trajectories into the Traj2Stroke model to get predicted stroke masks. These masks are stamped onto the before-stroke pictures, and the resulting prediction is compared to the after-stroke pictures. We minimize a weighted L1 loss that places higher weight on pixels covered by the new stroke. In practice, collecting the dataset (including setup and execution) takes around an hour, and training takes around 20 minutes to converge on our single-GPU system.

A separate Traj2Stroke model must be trained for each drawing medium (marker/brush), but is robust to out-of-distribution *trajectories*. This means that when we obtain a new TrajVAE, we can almost always plug-and-play it into the system without needing to collect new data and retraining Traj2Stroke. This is very convenient, as collecting the Traj2Stroke dataset is usually the most time-consuming part of preparations.

After receiving a standardized trajectory $[(x_1, y_1, h_1), \cdots, (x_n, y_n, h_n)]$ and pose offsets $\Delta = (\Delta_x, \Delta_y, \Delta_\theta)$, the Traj2Stroke model begins by reorienting the trajectory to be in the reference frame of the canvas (see Figure 5.3). To do this, it first rotates the x and y components by Δ_θ. Then, each rotated coordinate (x, y, h) is scaled and translated to become

$$(m_x x + b_x + \Delta_x,\ m_y y + b_y + \Delta_y,\ h)\,.$$

m_x, m_y, b_x, and b_y are learnable parameters used to model any small affine error that may occur during camera calibration. We expect that $m_x, m_y \approx 1$ and $b_x, b_y \approx 0$.

The trajectory has now been converted to canvas coordinates, and we denote it as

$$[(x'_1, y'_1, h_1), \cdots, (x'_n, y'_n, h_n)].$$

We proceed by rendering each of its $n-1$ segments separately. Fix an arbitrary k, and note that segment k goes from (x'_k, y'_k, h_k) to $(x'_{k+1}, y'_{k+1}, h_{k+1})$.

Our approach to rendering the segment is to first define a constant $H \times W \times 2$ tensor G of canvas coordinates, where H and W are the dimensions of the canvas. One channel of this tensor contains the x coordinates, and the other contains the y coordinates, as seen in Figure 5.3. For convenience, we also define $u = \begin{bmatrix} x'_k & y'_k \end{bmatrix}^T$ and $v = \begin{bmatrix} x'_{k+1} & y'_{k+1} \end{bmatrix}^T$.

We compute a *Distance Map* that stores the distance of each coordinate in G to the segment. This is computed with the following equation (note that the vector operations involving G are done element-wise):

$$\begin{aligned} \text{Distance Map} = \min(&\left\| (G-u) - \text{proj}_{v-u}(G-u) \right\|, \\ &\|G-u\|, \|G-v\|)). \end{aligned} \tag{5.1}$$

The first term computes the distance from each point in G to the line through u and v, and the last two terms calculate the distance to the endpoints. Thus, taking the minimum of the three yields the distance of each pixel to the line from u to v.

We also compute a *Height Map*, which represents the height of the brush tip as it moves over the segment. For each coordinate, we project it onto the segment and compute the height by linear interpolation between h_k and h_{k+1}:

$$T = \text{clamp}_{[0,1]} \left(\frac{\left\| \text{proj}_{v-u}(G-u) \right\|}{\|v-u\|} \right) \tag{5.2}$$

$$\text{Height Map} = (1-T) \cdot h_k + T \cdot h_{k+1}. \tag{5.3}$$

We approximate the relationship between the height of the brush tip and the thickness of the stroke as affine. Thus, we introduce two learnable parameters α and β, and obtain a *Thickness Map* like so:

$$\text{Thickness Map} = \alpha \cdot \text{Height Map} + \beta. \tag{5.4}$$

If the distance between a coordinate and the segment is less than the stroke thickness, then that coordinate should be affected by the stroke. We assume there is a gradual dropoff in darkness as we get further from the center of the segment. This reasoning motivates the following calculation for the darkness values:

$$\text{Darkness} = \left[\text{clamp}_{[0,1]} \left(1 - \frac{\text{Distance Map}}{\text{Thickness Map}} \right) \right]^c. \tag{5.5}$$

Coordinates directly on the segment get a darkness value of 1, and coordinates that are a stroke thickness away get a darkness value of 0. This also introduces another learnable parameter c which determines how quickly the darkness values drop off as they get further from the segment.

Finally, we take the max darkness values over all segments to obtain the rendered stroke.

In total, the Traj2Stroke model has only 7 learnable parameters: m_x, m_y, b_x, b_y, α, β, and c.

5.3.5 Stroke Composition

We define each brush stroke action as a set of parameters: TrajVAE latent vector z, pose offsets Δ, and RGB color ρ. Given z and Δ, Figure 5.3 illustrates how Spline-FRIDA predicts the shape of a single stroke. Next, the rendered stroke is colorized by duplicating it to 3-channels and multiplying each channel based on the stroke color as seen in Figure 5.5. We can predict how this stroke s will appear once it is performed on a canvas c_t by stamping it via an alpha blending formula.

5.3.6 Painting and Drawing Planning

To plan a painting or drawing, we follow the FRIDA [82] planning algorithm which plans paintings using an optimization loop, depicted in Figure 5.5. A user-specified number of brush stroke actions are randomly initialized. At each optimization step, the current canvas is compared to the user-specified target image forming a loss value. In practice, features from the planned painting and target image are extracted using pretrained neural networks (e.g., CLIP [67]) and compared using cosine similarity as introduced in [73]. The loss is back-propagated through the dynamics model to the brush stroke parameters which are updated using gradient descent.

If the robot is painting in color, the color parameters are optimized as continuous RGB values during initial iterations. In the last 10% of optimization iterations, the algorithm discretizes the colors to a user-specified number using K-Means clustering. After optimizing for 2000 iterations, the system shows which colors of paint need to be mixed in a graphical user-interface. The user mixes these paint colors, provides them to the robot, then the robot can begin painting.

5.4 Results

Figure 5.6 shows an array of drawings produced by Spline-FRIDA. We hand-pick five human drawings from members of our lab using the mocap system, each with distinct stroke styles, which are presented in the top row. Each human drawing is used to fine-tune a separate TrajVAE, resulting in five unique TrajVAEs. Each TrajVAE is then used to plan a series of drawings with various objectives. These objectives are displayed in the left column.

The individual styles of the drawing trajectories are preserved by the TrajVAEs. For instance, the fourth human drawing exhibits tiny, curly lines, which are reproduced in the drawings made using its corresponding TrajVAE. Similarly, the fifth human drawing is composed of small circles, which is also true for the robot drawings in its column.

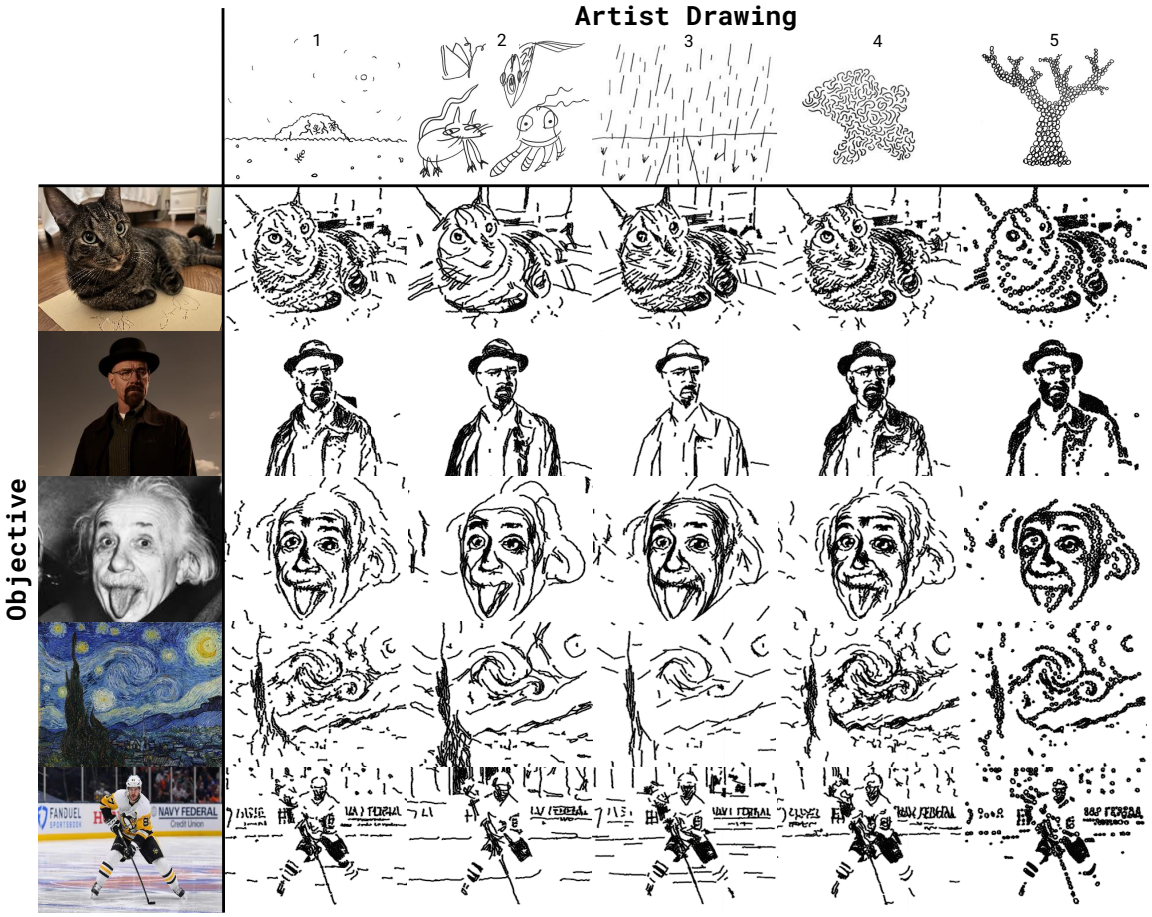

Figure 5.6: Example drawings made by Spline-FRIDA. Each column represents a distinct trajectory style and each row uses a different objective. The top row contains original drawings made by human artists on our mocap system. One VAE was fine-tuned on each human drawing and used to plan the drawings in each column.

5.4.1 Human Evaluations

To what extent is Spline-FRIDA able to capture the stroke style of a drawing? And, in general, are Spline-FRIDA's drawings better than those made by FRIDA? These questions are subjective and difficult to answer with automatic metrics. To obtain quantitative results, we conducted a survey and released it to 100 participants on Amazon Mechanical Turk.

For the first part of the survey, we asked participants to match Spline-FRIDA drawings with human drawings that have the same stroke style. More specifically, for each participant, we selected a random human drawing, along with five robot drawings (a random row of Figure 5.6), and asked them to pick the robot drawing that best matched the style of the human drawing. We told participants to "focus on the characteristics of individual strokes, such as their trajectories, shapes, and curves."

	FRIDA	Spline-FRIDA
Which drawing looks more like it was drawn by a human (rather than a robot)?	27	73
Which drawing looks better overall?	16	84
Which drawing better matches the reference image?	16	84
Which drawing is more artistic?	18	82
Which drawing is more abstract?	40	60

Table 5.1: Opinions on FRIDA vs Spline-FRIDA. Each cell shows the number of participants that chose the system for the given question. Overall, participants thought that compared to the Bèzier curve representation of FRIDA, drawings made by Spline-FRIDA were more human-like, higher quality, more true to the objective, and more artistic.

The results of this experiment are seen in Figure 5.7.

The high values of Figure 5.7 along the diagonal suggest that, in general, participants were able to choose the correct human drawing used to style each robot drawing. Style 5 seemed to be particularly distinguishable. Meanwhile, style 1 was often confused with style 4, and style 2 was confused with style 3. Nevertheless, all five encoded styles are most strongly associated with the correct human drawings.

The second part of the survey asked participants' subjective opinions on Sharpie drawings made by FRIDA vs. Spline-FRIDA. Each participant was shown an objective image and two robot drawings of it, one from FRIDA and one from Spline-FRIDA. Both robot drawings were executed on the physical robot so that any Sim2Real gap comes into play. The questions and tallied responses are shown in Table 5.1.

Respondents believed that Spline-FRIDA's drawings, in comparison to FRIDA's, appeared more human-made, had higher overall quality, better matched the reference image, and were more artistic. Respondents also perceived the Spline-FRIDA drawings as more "abstract", although opinions on this were somewhat split.

5.4.2 Trajectory Distributions

In Figure 5.8, we visualize the latent space for our TrajVAE trained on multiple drawing sessions. We encode all of the human trajectories into latent vectors, then project them down to 2 dimensions using t-SNE [93]. We then draw each human trajectory at its corresponding 2d coordinates. We observe human trajectories spread throughout the space, forming several homogeneous clusters. This structured organi-

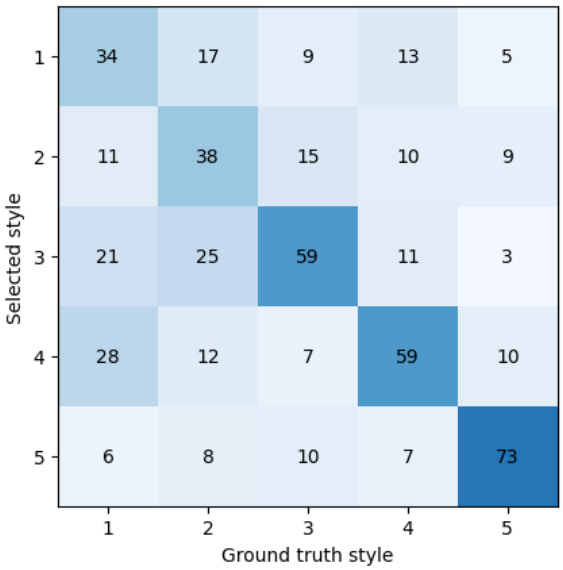

Figure 5.7: Confusion matrix for matching task. The x-axis represents the index of the specific TrajVAE used to generate the drawing, and the y-axis represents the index of the human drawing participants thought was most similar. The five human drawings/styles the same ones as in the top row of Figure 5.6, with the same order.

Figure 5.8: Mapping the latent space. We visualize the TrajVAE latent space by drawing trajectories at their respective coordinates, projected down to 2 dimensions via t-SNE. To generate this plot, we use a TrajVAE that is trained on multiple sessions of human trajectory data.

zation indicates that TrajVAE effectively learns a correlation between trajectories and latent vectors. Consequently, an optimization-based planning algorithm is likely to be

effective.

5.4.3 Brush Stroke Dynamics Modeling Experiments

The purpose of a stroke dynamics model is to differentiably render trajectories. We experiment with four different methods and evaluate them both quantitatively and qualitatively in a controlled experiment.

CNN

Our baseline, a convolutional neural network starting with a fully connected layer and followed by several transposed convolutions. This is analogous to FRIDA's [82] renderer architecture, but takes full trajectories rather than the parameterization (length, bend, height) that FRIDA uses.

CNN with CoordConv

To render a trajectory, one subproblem the renderer must solve is mapping Cartesian coordinates to one-hot pixel space. Liu et. al. [94] showed that traditional CNNs can have difficulty with this, so we implement their suggestion of using CoordConv layers instead of traditional convolutions. This means adding two additional channels to the input of each convolution: one containing the x-coordinates of each pixel, and the other containing the y-coordinates.

Traj2Stroke

This is our main method, with the rule-based transformations, that was described in Section 5.3.4.

Traj2Stroke with U-Net

Our main method, but with an additional convolutional network attached after the output layer. The goal of this additional network is to refine the Traj2Stroke output by learning subtle effects such as texture and bristle drag. Its architecture closely follows that of U-Net [95]. We freeze the U-Net weights during the first half of the training and unfreeze them for the second half. The purpose of this is to train the Traj2Stroke portion first and get it as close as possible to the ground truth, before using the U-Net to refine it. Inspired by the success of ControlNet [96], the U-net is initialized with a zero-convolution final transformation so that it initially performs the identity function.

Since generating training strokes and training a new dynamics model for every new stroke style is time consuming, the ability of the dynamics model to generalize to unseen styles is important. In order to evaluate generalizability, we train and test the

Medium	CNN	CNN w/ CoordConv	Traj2Stroke	Traj2Stroke w/ U-Net
Sharpie	.00107	.00095	**.00055**	.00098
Brush	.00162	.00163	.00158	**.00153**

Table 5.2: Quantitative comparison of stroke models. This table shows the average L1 loss of each stroke model when predicting either sharpie or brush strokes (lower is better). Loss is calculated on dataset B (out-of-distribution) trajectories only. Traj2Stroke achieves the best results for sharpie strokes, and Traj2Stroke with U-Net is the best for brush strokes.

stroke model on trajectories from different distributions. More precisely, we create two datasets, A and B. Both datasets contain (trajectory, stroke image) pairs. For dataset A, the trajectories are sampled from a generic TrajVAE, trained on a session that we judge to have good stroke diversity. For dataset B, we use trajectories from more specialized TrajVAEs, trained on sessions with very unique styles. We train the model using dataset A, and we evaluate generalizability by checking its performance on dataset B.

We run the experiment twice, once for each of two drawing mediums: a sharpie and a thin paintbrush. The experiment results can be seen in Table 5.2. The Traj2Stroke architecture without U-Net achieves the lowest loss on sharpie strokes. Adding the U-Net hurts performance on sharpie strokes, though it achieves the best results on brush strokes. There is not a substantial increase in performance from using CoordConv over traditional the pure CNN architecture.

Visually, example predictions generated by each model can be seen in Figure 5.9. All examples are from dataset B, meaning that these trajectories are out of distribution from the training set. The vanilla CNN with and without CoordConv fails to generalize in certain cases. The Traj2Stroke model performs near-perfect for the sharpie strokes and captures the general shape of the brush strokes. Adding the U-Net to Traj2Stroke helped capture the texture of the brush strokes, but the added parameters hurt generalization to very out-of-distribution strokes, such as the star example.

Based on these findings, we choose to implement the base Traj2Stroke model (without U-Net) for Spline-FRIDA. As illustrated in Figure 5.10, the resulting Sim2Real gap for Sharpie drawings is very low. This is a huge improvement compared to the original FRIDA results depicted in Figure 5.10.

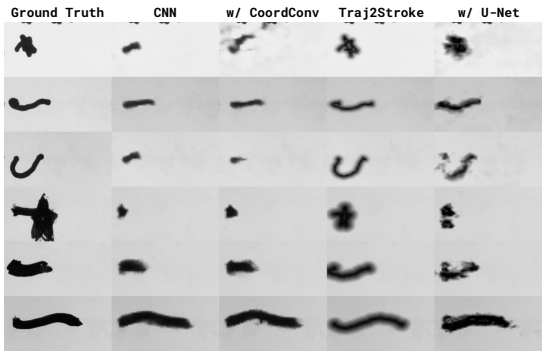

Figure 5.9: Visualizing the outputs of various stroke models. The first three rows contain sharpie strokes, and the last three contain brush strokes all of which were made from samples using TrajVAE models not used for training.

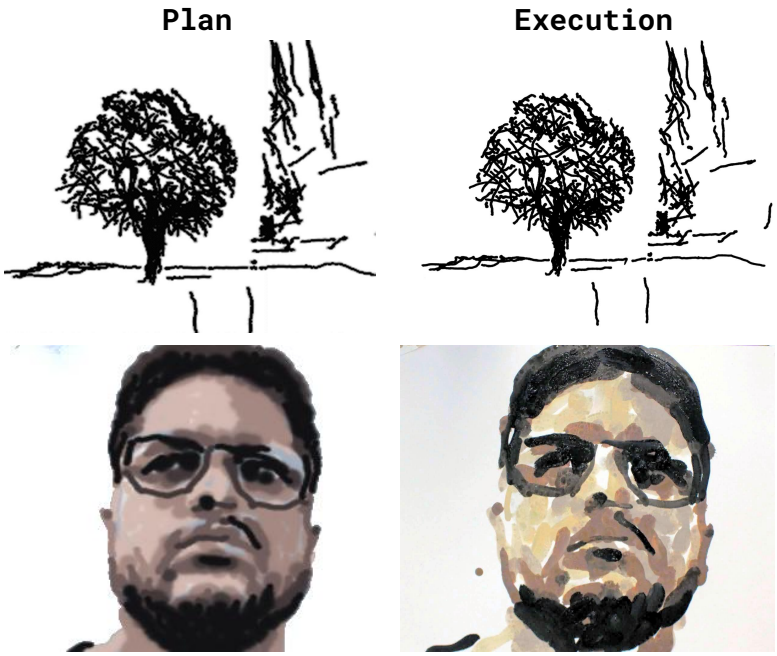

Figure 5.10: Spline-FRIDA's low Sim2Real gap. We compare a plan made by Spline-FRIDA with its execution (physically drawn with a robot). The top row with a black marker, and the bottom row with a paint brush.

5.5 Conclusions

In Chapter4, we showed that our dynamics model was able to accurately model brush strokes with zero prior information. This allowed the model to learn complex details about the brush and paint without the hindrance of prior rules written by designers. However, without a prior, in this chapter, we showed that dynamics cannot

be modeled accurately with a feasible number of training examples when the action representation becomes more expressive. Our Spline-FRIDA model adds a prior to the rendering algorithm that allows it to learn the dynamics of strokes that are as complex as human demonstrators can produce.

FRIDA is capable of learning to paint without any demonstrations or prior knowledge of how humans paint. We argue that a learning from demonstration (LfD) framework for painting would require millions of demonstrations to train, which is infeasible. Although LfD would be infeasible to learn the entire painting task, it is not necessarily useless for painting. In this chapter, we showed that LfD can be used within the FRIDA framework to improve the action representation to make the brush stroke trajectories more human-like. Our human evaluation study showed that the LfD stroke shapes were more human-like than the Bézier curves. Interestingly, these more human-like strokes helped make drawings that were perceived as better looking, more artistic, and also more abstract. More experiments are needed to fully understand what caused the improved appearance, but this work shows that having more natural actions in a robot can improve its performance in a number of categories.

III

Supportive Goal Planner: *What* the Robot Paints

6 Collaborative Goal Creation for Robot Painting

6.1 Introduction

In this chapter, we introduce the Supportive Goal Planner within the FRIDA painting system. The goal of the Supportive Goal Planner is to plan *what* the robot should generate. This can then be passed to the Low-Level Action Planner for the robot to figure out *how* to paint it. In the context of painting and drawing, these generated plans are RGB image previews of what the canvas should look like after the robot is done. We adopt a collaborative form of support in this chapter, where the robot takes turns with a person to add to a drawing or painting. The generated plans should (1) fit the input text descriptions that the user gives, (2) use the current canvas state without overwriting everything there already, and (3) be achievable by the robot's tools, materials, and actions. We call the task of satisfying these three constraints *co-painting* defined as a form of human-robot co-creation where the robot creates new content that engages with the existing content that the human drew or a robot drew previously.

While there exist related image editing problems, such as in-painting, co-painting is a new class of problems with unique challenges as it is undesirable in co-painting to make radical changes to the image that would overwrite the human's previous work. In in-painting, the area for editing an image is coarsely specified by the user and the model is expected to drastically change the content within that local region. By contrast, with co-painting, the edit is expected to preserve and engage with the full canvas rather than re-imagining a local region. Whereas in-painting is a localized edit by definition, co-painting is a continuous, iterative completion, e.g., adding detail to an existing human-drawn rough sketch.

Besides the challenges of co-painting, robotic image creation is difficult due to real-world constraints, such as existing canvas state, limited abilities of the robot, tools and materials available to the robot, and stochasticity in robot performance. These robotic constraints vastly limit the content that is capable of being created, as illustrated in the left side of Figure 6.8. With a large paintbrush, fine-details

are not achievable, and with a single marker, multi-color images are not possible. Multiple works address these constraints to decrease the Sim2Real gap, but only paint from image inputs [15, 16, 61, 66]. Even fewer existing works use cameras to enable co-creation of images [97].

The Low-Level Action Planner presented in Chapter 4 is capable of planning paintings to match text prompts and planning on an existing, non-blank canvas. To paint from a language input, the Low-Level Action Planner uses CLIP [67] to align language and image which tends to generate noisy output. To improve the quality of paintings for FRIDA, we introduce the Supportive Goal Planner which adapts powerful image generators pre-trained using gigantic text-image paired data, e.g., StableDiffusion [4] or Instruct-Pix2Pix [98]. Because such pre-trained image generators do not know the capabilities of the robot, there is both a large difference in pixel value and semantic meaning between the image generator output and Low-Level Action Planner's simulated plan. The former difference is a traditional Sim2Real gap, whereas the latter is a concept we introduce as the *Semantic Sim2Real Gap*.

Figure 6.1: Co-Painting with FRIDA. We showcase how FRIDA with the High-Level Planner can collaboratively paint with artists. The process begins with the artist sketching a table. Building on that foundation, FRIDA adds to the canvas, guided by the artist's initial prompt: "A bulky robot arm on a table." The artist then iterates on the painting with additional strokes to add detail to the robot arm, and provides a new text prompt, "A robot arm with a hand." FRIDA responds by completing the painting to match this new description.

To reduce the Semantic Sim2Real Gap, we propose a Self-Supervised Fine-Tuning approach for the Supportive Goal Planner. Self-Supervised Fine-Tuning adapts a pre-trained image generator to both generate content within the abilities of the robot and perform co-painting to enable human-

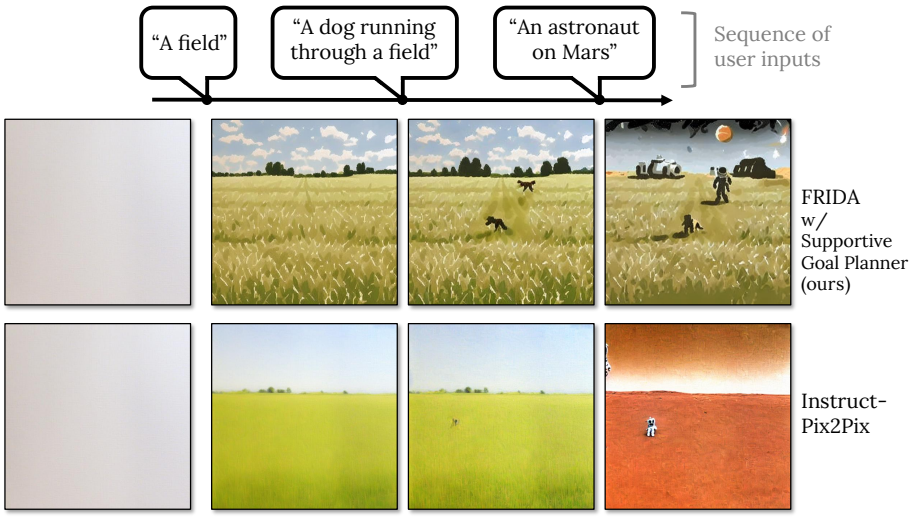

Figure 6.2: Co-Painting. We introduce Co-Painting as a task in which a robot must add content to a painting that engages with the current content without destroying the existing work. We demonstrate that existing models (Instruct-Pix2Pix, bottom row) often cannot successfully add content without making unreasonably large edits to the canvas, overwriting any prior work, while FRIDA (top row) adds content that harmonizes with the existing work.

robot collaborative drawing from language guidance, e.g., in this paper, we use Instruct-Pix2Pix [98] as our base text-image model. To adapt a pre-trained model for co-painting and encode robotic constraints, first we create the self-supervised fine-tuning dataset by using the Low-Level Action Planner to generate full drawings or paintings of images from a text-image dataset. Strokes from the full paintings are removed selectively to form partial paintings. We fine-tune Instruct-Pix2Pix by retraining it with a low learning rate to predict the full painting from the partial painting and text prompt.

The Supportive Goal Planner can successfully use an existing canvas state to generate future actions towards a language goal without completely overwriting the existing work as shown in Figure 6.2. Based on a survey on Amazon Mechanical Turk (MTurk) of 24 participants, FRIDA with the Supportive Goal Planner's completed drawings from partial sketches were found to be substantially more similar to the language goal when compared to those by the baselines.

6.2 Motivation

While recent breakthroughs in text-to-image synthesis technologies have ignited a boom in digital content generation, using them to produce art with robots is still in

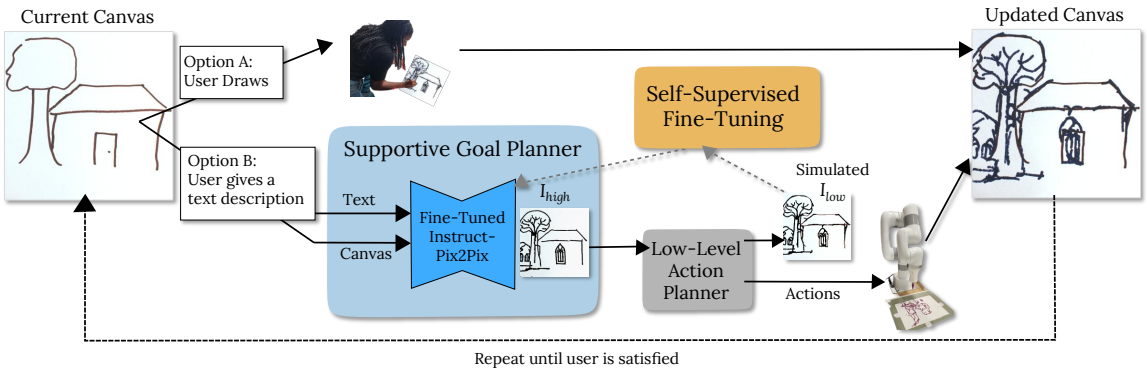

Figure 6.3: Method Overview. Offline, we fine-tune a pre-trained Instruct-Pix2Pix model on our self-supervised data. Online, the user can either draw or give the robot a text description. The Supportive Goal Planner takes as input the current canvas and text description to generate a pixel prediction of how the robot should finish the painting using the fine-tuned Instruct-Pix2Pix model. The Low-Level Action Planner predicts actions for the robot to create this pixel image and produces a simulation. This process is repeated until the user is satisfied.

its infancy due to a significant gap between simulated and real-world environments. FRIDA with just the Low-Level Action Planner is capable of generating paintings from image or text inputs, but its interaction is limited to the input stage after which the robot paints without additional input by the user. While it is still debatable whether such an autonomous creation is desired by humans practicing art [1], there is strong evidence of the potential value of a co-creative agent [2, 78, 79, 80, 81, 99, 100] specifically in the domain of art therapy [101, 102, 103, 104]. The benefits can be further increased when paired with a physical embodiment of such an agent and drawing in the real world [97, 105]. To invite users into the creative process and bring the benefits of both co-creation and robotic embodiment, we build on the Low-Level Action Planner to propose a Supportive Goal Planner into the FRIDA system which can collaborate with a human, as illustrated in Figure 6.1.

6.3 Related Work

6.3.1 Computer-Based Image Co-Creation

Computer-based image co-creation generally involves turn taking between a human and a computer in applying brush stroke primitives towards one of a discrete set of goals, as in `sketch-rnn` [106] and Drawing Apprentice [79], or even towards natural language goals [80, 99]. Computer-based studies have shown creativity augmentation benefits of co-creation [79, 80, 99] since computer agents can add serendipity and reformulate user's original intentions leading to unexpected by enjoyable outputs [100]. However, Computer-based painting models do not transfer well out-of-the-box into

the real world due to the Sim2Real gap [18, 61, 107].

6.3.2 Robotic Image Co-Creation

There exists many real-world methods for robot painting and drawing [15, 16, 28, 62], however, few systems have incorporated perception into their systems to enable co-painting. Cobbie [97] is a co-drawing system that boosted ideation for novice drawers, however, it is limited to drawing on blank areas of the paper rather than engaging with the user drawn content. [104] created a robot arm that can draw from speech inputs that are limited to simple objects found in the Quick, Draw! dataset [108].

The Low-Level Action Planner (Part II) is capable of making paintings that use the existing content conditioned on natural language goals by itself without the Supportive Goal Planner. Figure 4.12 shows examples of text-conditioned paintings using just the Low-Level Action Planner. While the Low-Level Action Planner can plan based on current canvas state and language input, it uses CLIP and gradient descent for planning which produces paintings that are very noisy and only loosely resemble the input text.

6.4 Approach

Shown in Figure 6.3, our method for co-painting, FRIDA, is made up of three primary components: (1) The Supportive Goal Planner, which produces images illustrating how the robot should add content to an existing canvas given a text description, (2) the Low-Level Action Planner, a robotic painting system for planning actions from given images, and (3) a self-supervised method for creating training data using the Low-Level Action Planner to fine-tune pre-trained models in the Supportive Goal Planner.

6.4.1 Self-Supervised Data Creation

While there exist some supervised data of human-created co-paintings [109, 110], they are only on the order of tens of examples and were not made using the same materials available to our robot. To support co-painting tasks, we propose a self-supervised method for generating training data to train the Supportive Goal Planner. We simulate paintings of images from the art subset of the CoCo image-text dataset [111] using FRIDA with image-guidance loss (difference of CLIP embeddings of images). To create partial paintings, strokes are removed selectively to support a variety of co-painting tasks: remove all strokes, a random subset of strokes, strokes corresponding to a salient region (defined with CLIP as in [73]) of the image, and strokes from a semantic region (using Segment Anything [112]). Illustrative examples are shown in Figure 6.4.

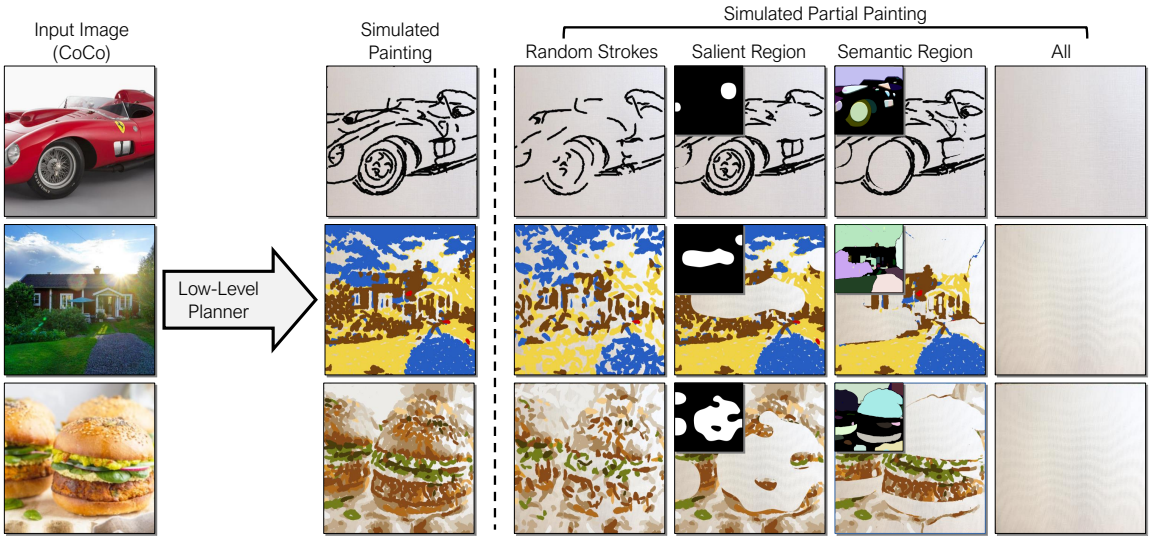

Figure 6.4: Self-Supervised Dataset Creation. We describe the process of generating the self-supervised training data pairs for fine-tuning the Supportive Goal Planner. We start with the input images from the CoCo dataset and convert them into simulated sketch outputs with the Low-Level Action Planner's simulator. Next, we create partial sketches in four different ways: removing random strokes, removing the salient region, removing a semantic region, and removing all strokes.

Some source images cannot be accurately represented with the robot's abilities. We filter out such images by removing instances that have a CLIPScore between the simulated full paintings and the text less than 0.5.

We use this self-curated data to fine-tune a base text-to-image generation model to be able to 1) continue to create content on an existing canvas and 2) generate images that the target robot is capable of painting.

6.4.2 The Supportive Goal Planner

The goal of the Supportive Goal Planner (Figure 6.3) is to generate an image of how the robot should complete the painting given a photograph of the current canvas and a user given text description. The Supportive Goal Planner uses Instruct-Pix2Pix [98] as a pre-trained model as it enables conditioning the output on an input canvas. The pre-trained Instruct-Pix2Pix, however, has two shortcomings to be used for co-painting: (1) the generated images do not reflect actual robotic constraints, and (2) the existing canvas can sometimes be overwritten completely as shown in Figure 6.2. To overcome these limitations, we fine-tune Instruct-Pix2Pix using the dataset of partial and full drawings with their captions described in Sec. 6.4.1.

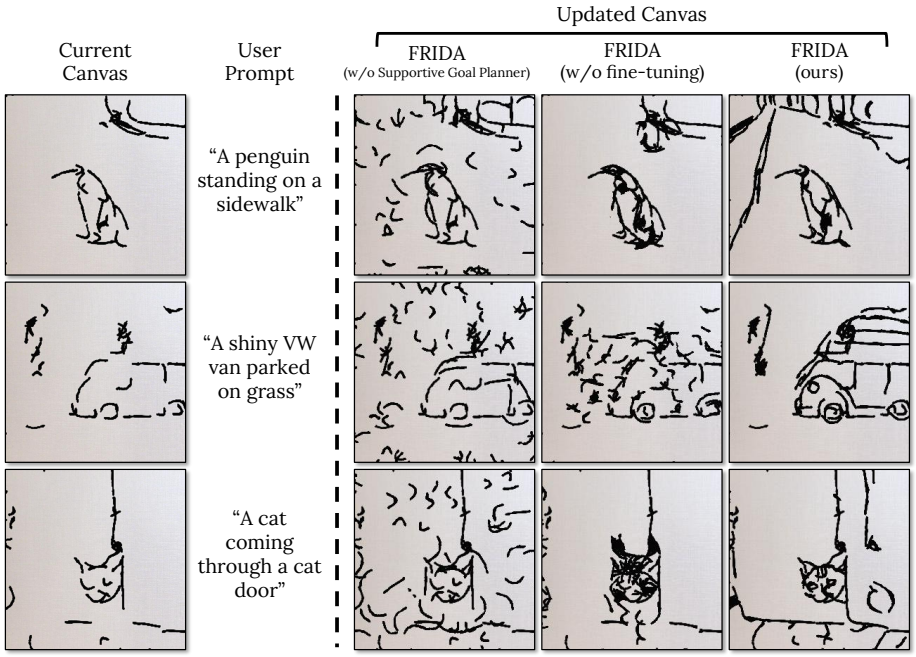

Figure 6.5: Qualitative Comparison. We show a comparison between three methods of performing text-based canvas updates: FRIDA without the Supportive Goal Planner (using the text-based Low-Level Action Planner only), FRIDA without fine-tuning the Supportive Goal Planner, and FRIDA with self-supervised fine-tuning (ours). FRIDA with just the text-based Low-Level Action Planner uses a CLIP based optimization and generates outputs that are noisy. FRIDA without fine-tuning, is not aware of the constraints of the robot and generates an output that is difficult for the robot to execute and often does not satisfy the text prompt specified by the user. In contrast, FRIDA outputs an updated canvas that reflects the user prompt without being noisy.

Fine-tuning is performed using the Low-Level Action Planner's simulated canvases because (1) it would be infeasible to generate a large-scale dataset with the physical robot, and (2) the Supportive Goal Planner output is eventually used with the Low-Level Action Planner's simulation.

6.5 Experiments

6.5.1 Baselines

We compare FRIDA with the Supportive Goal Planner versus FRIDA with just the Low-Level Action Planner using the CLIP-guided text-to-painting method (FRIDA w/o High Level Planner). We investigate the effects of our fine-tuning procedure on Instruct-Pix2Pix in the Supportive Goal Planner by comparing our method (FRIDA) with pre-trained Instruct-Pix2pix (FRIDA w/o fine-tuning).

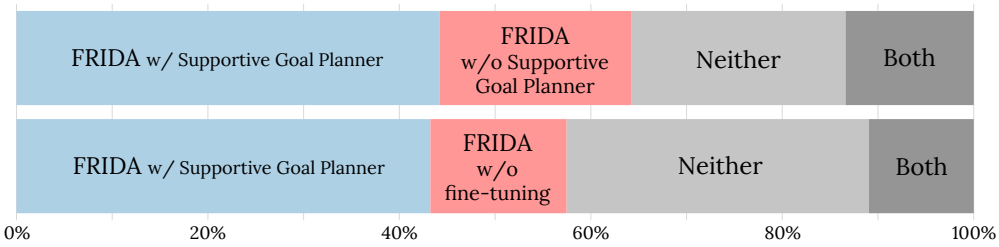

Figure 6.6: User Preference Study. Results from two MTurk Surveys. Presented with a text description, participants chose which of two drawings (FRIDA versus either FRIDA without the High-Level Planner or FRIDA without fine-tuning) was more similar to the text, neither, or both. See Fig. 6.5 for examples.

	CLIPScore ↑	BLIPScore ↑	Δ_{pix} ↓	Δ_{sem} ↓
FRIDA w/o Supportive Goal Planner	**0.741**	**0.192**	—	—
FRIDA w/o fine-tuning	0.595	0.162	0.195	0.241
FRIDA	0.624	0.178	**0.052**	**0.035**

Table 6.1: CLIPScores and BLIPScores computed on robot simulated drawings (See Figure 6.5). Sim-to-real gap measurements, Δ_{pix} and Δ_{sem}, measure the difference between the Supportive Goal Planner output and the simulated drawing of that image.

6.5.2 Different Painting Settings

The Low-Level Action Planner can paint and draw with various brushes and can have different color constraints. We test FRIDA using three different painting settings (1) acrylic painting using one brush and 12 colors which can differ from painting to painting, (2) acrylic painting with a fixed 4-color palette, and (3) a black Sharpie marker. Examples of these three settings are shown in Figure 6.4, 6.9, and 6.8 . The robot can only be used in one of these settings at a time. However, users can paint using any media of choice, leading to mixed media paintings in Figure 6.7.

6.5.3 Evaluation

Text-Image Alignment - Two automatic methods of comparing image and text are CLIPScore [113] and BLIPScore [114], which measure the similarity between images and text with a pre-trained image-text encoders. Because FRIDA's Low-Level Action Planner (FRIDA w/o Supportive Goal Planner in Table **??**) with text

guidance directly optimizes the CLIPScore to create images from text, this method is unfairly advantaged when using CLIPScore. We use MTurk to achieve large-scale fair evaluation of text-image alignment.

Semantic Sim2Real Gap - It is important that between the output of the Supportive Goal Planner (I_{high}) and the Low-Level Action Planner's simulation (I_{low}) there is little loss in semantic meaning. A naive approach at measuring this loss is the mean-squared-error between the images' pixels (Eq. 6.1). However, this is sensitive to low-level variation in details such as color or tone differences which are tolerable as long as the high-level content in the images is the same. To measure the high-level difference, we propose to use the cosine distance between CLIP image embeddings, Δ_{sem}, Eq. 6.2, referred to as the Semantic Sim2Real Gap.

$$\Delta_{pix} = ||I_{high} - I_{low}||_2^2 \tag{6.1}$$

$$\Delta_{sem} = \cos(CLIP(I_{high}), CLIP(I_{low})) \tag{6.2}$$

A proper Sim2Real gap measurement would compare the output of the Supportive Goal Planner to the real drawing, however, it is infeasible to generate a robust number of real-world samples. Because the Sim2Real gap between the Low-Level Action Planner's simulation and the real drawing is the same across all tested methods, we can fairly use the FRIDA simulations in lieu of the real drawings for comparing the Sim2Real gaps of Supportive Goal Planner variations.

6.6 Results

6.6.1 Co-Painting

To test the ability of FRIDA to work with an existing canvas state, we focus on Sharpie marker drawings where no erasing is possible, forcing the model to have to adapt to and use the existing markings on the page. To create the partial drawing, we generate an image with Stable Diffusion using prompts from the PartiPrompts [115] dataset, then simulate the drawing with just 35 strokes as depicted in Figure 6.5. We generated 40 images from different prompts per method. CLIPScore [113], BLIPScore, and Sim2Real gap measures are reported in Table **??**. Since the text-guided Low-Level Action Planner maximizes CLIPScore, it was expected and confirmed that FRIDA without the Supportive Goal Planner has the highest CLIPScore. BLIP is also expected to correlate with CLIP, leading FRIDA without the Supportive Goal Planner to have an artificially high BLIPScore.

To properly assess the image-text similarity of the drawings from partial sketches, we conducted an MTurk survey summarized in Fig 6.6. 24 unique participants were shown a language description then two images (one from ours and the other one of the two baselines, in random order). Participants were instructed to choose which image fits the given caption better, or to select neither or both. Each image pair was

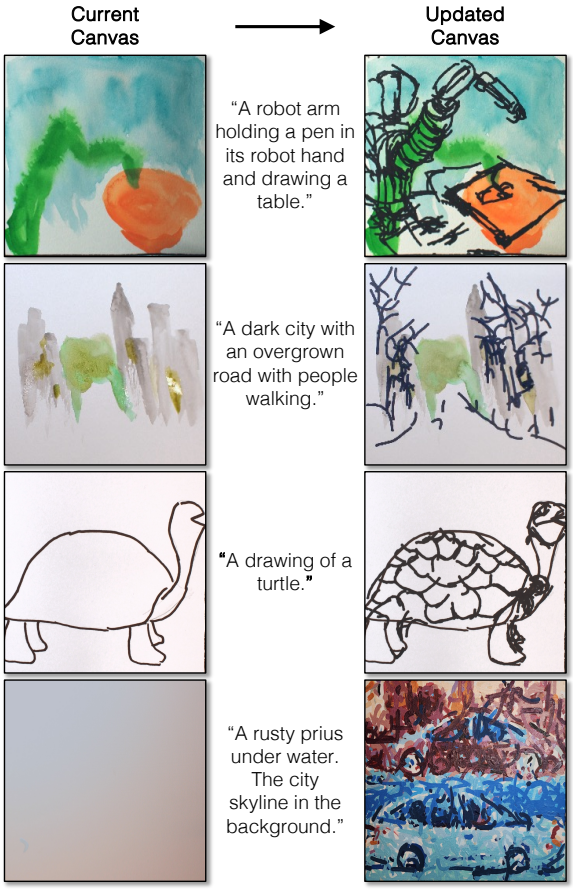

Figure 6.7: Mixed-Media Paintings. FRIDA can use markers and paintbrushes to co-paint with a human. Despite being fine-tuned with a single medium, FRIDA can still perform co-painting when a user uses different media such as watercolors.

evaluated by 4 unique participants leading to 160 comparisons per baseline. While many participants found neither image fit the text description (an indicator of the challenging nature of co-painting), FRIDA was generally indicated as having clearer content over FRIDA without the Supportive Goal Planner and FRIDA without our fine-tuning.

In terms of the proposed Semantic Sim2Real Gap, FRIDA outperforms the baselines indicating that our fine-tuning guided Instruct-Pix2Pix to produce images that were less likely to change meaning when planned by the Low-Level Action Planner.

6.6.2 Multiple Turns

A co-painting system must be capable of accommodating multiple iterations of human-robot interaction in which the robot adds content but does not completely

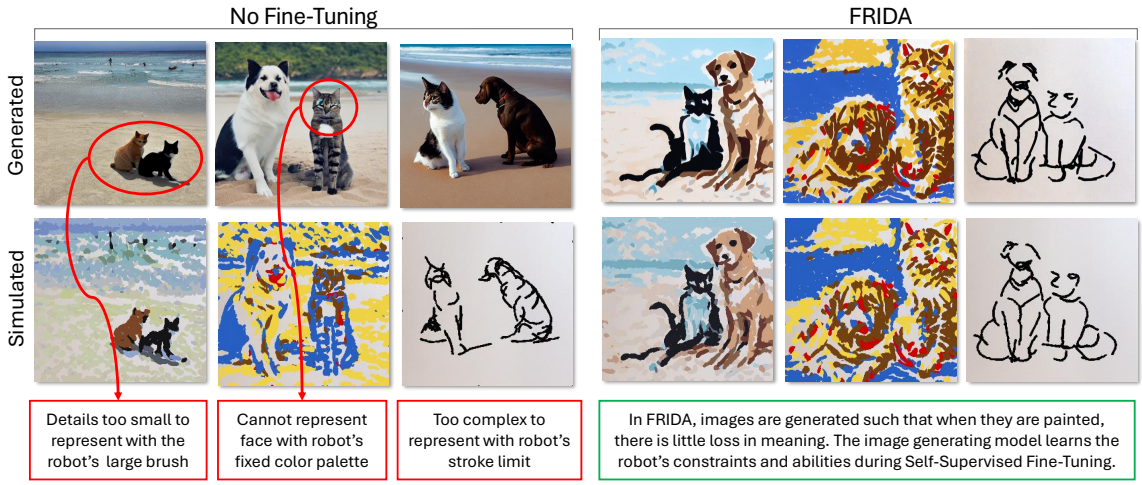

Figure 6.8: Learning Robotic Constraints. We compare images generated by a pre-trained Stable Diffusion model (left) to those generated by our proposed FRIDA's High-Level Planner (right) with the prompt "A dog and a cat sitting next to each other on the beach" in three different painting settings (Sec.6.5.2). The top row shows the images generated by each of the models and the bottom row shows the corresponding FRIDA Low-Level Planner simulation.

overwrite the human's prior work. We simulate this by having the robot create sequences of modifications to a simulated painting with different text prompts in Figure 6.2. The baseline methods tend to either avoid making changes or make huge changes to the canvas, whereas FRIDA makes updates that are more reasonable for the robot to achieve and integrate naturally with prior work.

6.6.3 Text Conditioned Paintings

The text-guided Low-Level Action Planner method relies on feedback through CLIP which results in noisy, unclear imagery. We compare FRIDA with the Supportive Goal Planner which uses a pre-trained generative model to text-guided Low-Level Action Planner in Figure 6.9. The Supportive Goal Planner creates paintings which are far more clear and capture the caption better than the text-guided Low-Level Action Planner in various painting settings.

6.6.4 Real Paintings

We used FRIDA's simulation to make large scale data creation and evaluation feasible. Figure 6.7 displays multiple real-world examples of FRIDA's drawings and paintings. FRIDA is able to successfully use content on canvases that is out of distribution from its fine-tuning training data as with the watercolor and marker

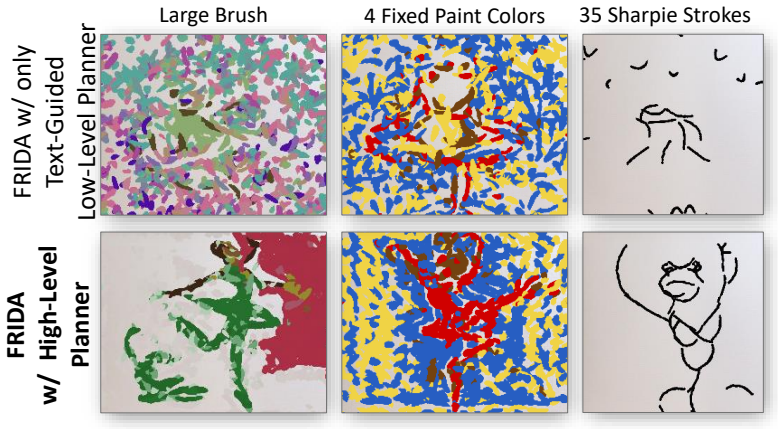

Figure 6.9: Comparing FRIDA's fine-tuned pre-trained image generator versus FRIDA's CLIP-guided method for generating paintings from the text "A sad, frog ballerina doing an arabesque" in three painting settings. Comparing FRIDA with the Low-Level Action Planner's text-guided abilities versus FRIDA with the Supportive Goal Planner. All paintings generated from the text input "A sad, frog ballerina doing an arabesque" in three painting settings.

examples in Figure 6.7.

6.7 Robot Synesthesia

To support richer interactions with the robot than just text inputs, in Robot Synesthesia [116], we introduced sound interactions to the FRIDA system to guide the generated painting. We treated natural sounds and speech separately. Natural sounds such as a horse neighing or laughter can guide the content of the painting (Figure 6.10). There is so much nuance to speech besides the words, so in addition to the text transcribed from speech audio, emotion predicted from the speech is also used to guide the painting. In Figure 6.11, the effect of emotional guidance can be seen. The motivation for this work was to design a robot that not only does what the user wants it to do, but also hears them, understands their emotions, and assists

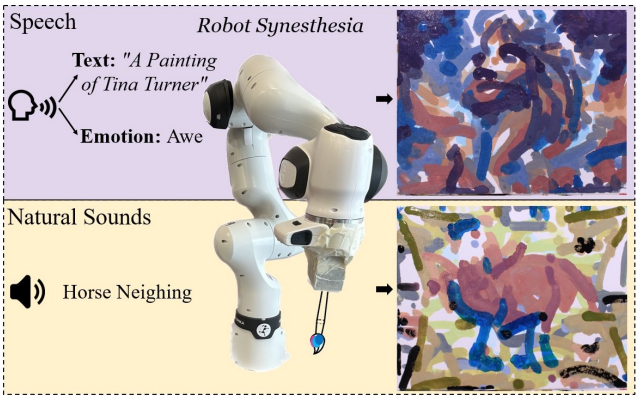

Figure 6.10: Robot Synesthesia - In this work, we added speech and sound guidance into the FRIDA system. Speech was decoupled into text and emotion.

them to express their ideas in visual art.

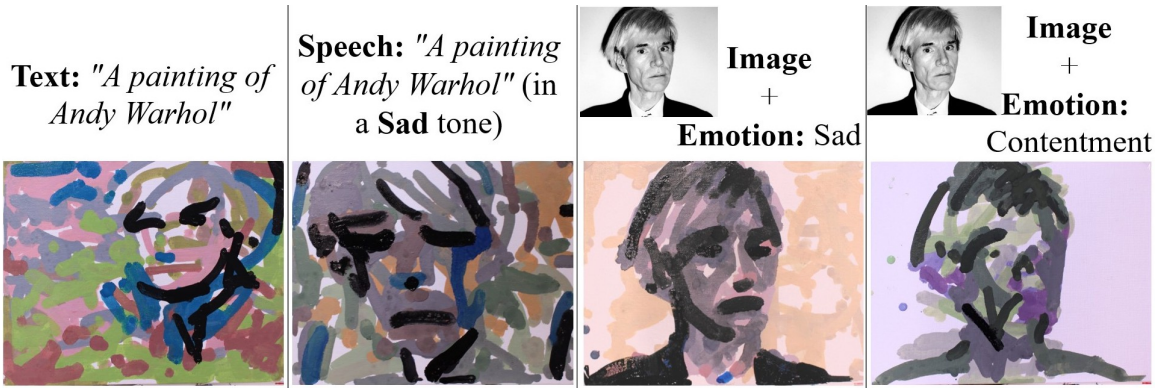

Figure 6.11: Speech is nuanced and more than just the words said. In Robot Synesthesia, the emotion from a given speech input is also used to guide the painting. Emotion can also be used with image inputs to add moods to existing content.

6.8 Discussion

6.8.1 Limitations and Ethical Considerations

FRIDA stands out as a successful collaborative painting system, but is limited to discrete turn-taking interactions. While our self-supervised training data creation method (Figure 6.4) was informed by real co-painting data, a more end-to-end approach where the system learns how to form the partial paintings could result in even better results.

FRIDA is subject to the biases of Stable Diffusion [4] and its training data [117], and so we recommend the usage of FRIDA with caution and solely for research purposes.

6.8.2 Learning Robotic Abilities

Our self-supervised fine-tuning procedure guided the pre-trained model to generate images that, at a pixel-level, appeared similar to what FRIDA can paint, but is it learning the actual robot constraints or just a low-level style transfer? We computed the Sim2Real gap measurements between the CoCo images and their FRIDA simulations (as seen in Figure 6.4) along with the CLIPScore of the simulation and text prompt. We found that Δ_{pix} had a small, insignificant Pearson correlation (-0.08, 0.08 p-value) with the CLIPScore of the painting whereas Δ_{sem} had a significant, negative correlation (-0.48, $2.4e-31$ p-value). Because our Self-Supervised Fine-Tuning scheme greatly decreases the Δsem, this indicates that our fine-tuning technique is not solely changing

the low-level appearance (akin to style-transfer) over the output of its base model. It appears that Self-Supervised Fine-Tuning is encoding the robot's abilities into the image generator, as seen in Figure 6.8 where FRIDA's Supportive Goal Planner produces images with (1) very prominent and clear content, when the robot's brush is large (2) select and limited colors, when the robot paints with fixed palettes or markers, and (3) sparse, concise drawings when the number of strokes is limited.

6.9 Conclusions

Robot capabilities need to meet the desires of humans who will use them and be affected by them. Our research did not find any desire to automate the artistic process [1], instead we found great evidence for the need for collaborative creative tools [2]. Supporting collaborative painting (and many collaborative tasks in general) is challenging because there are very few datasets that record this type of interaction. In this chapter, we showed how to enable the robot to create its own training data to train a collaborative goal creator. We train a foundation model base to (1) be collaborative by using the existing state towards the user's goal and (2) understand the abilities and constraints of the robot. The methodology here can be used to make a model predictive control system more collaborative by giving it a goal creation module.

IV

Generalization to Sculpting

7 Visual Sculpting

In Parts II and III, we showed our approach applied to the Generative Robotics task of collaborative painting. To show that our approach generalizes to other Generative Robotics tasks, in this part we apply our approach to robot sculpting. We show that our approach generalizes to new state representations (depth maps), materials (clay, foam, and sand), tools (grippers and end-effectors), and action representations (pinches and pokes).

7.1 Introduction

Sculpting has a rich history of expressing artistic meanings through 3D forms that captivate our sights, sense of touch, and emotions. Clay sculpting is a long-horizon, dexterous task where an artist takes a sequence of actions to modify the clay until the visual form is aligned with their underlying intentions. In this article, we formulate the

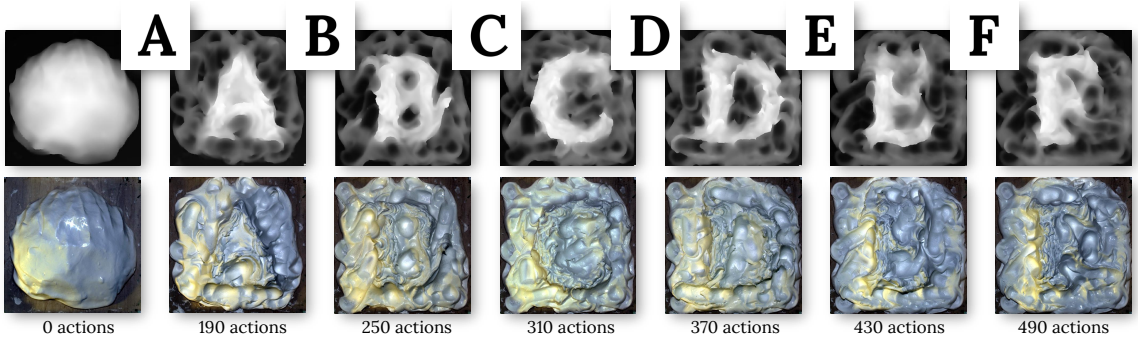

Figure 7.1: Long-Horizon. We tested our system's ability to perform long-horizon planning by sculpting the alphabet without resetting the clay between goals. The top row displays the goal images followed by depth maps and photographs of the real sculpted clay along with the total cumulative actions.

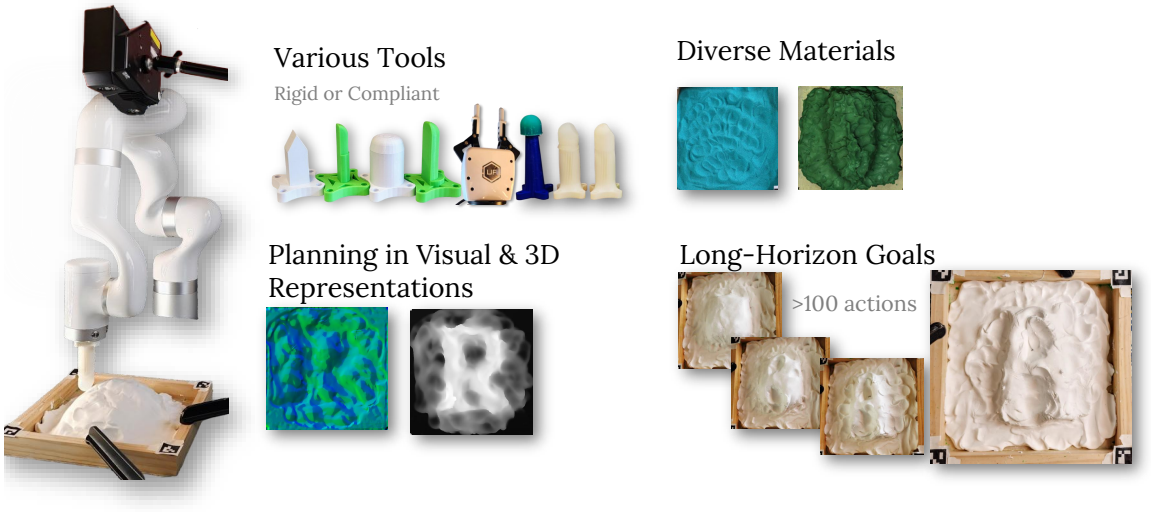

Figure 7.2: Visual Robotic Sculpting. We propose an approach to robotic sculpting that models deformable material dynamics in dense, high-resolution depth maps but plans in both 3D and visually-aligned representations in order to more closely align with human perception of 3D objects.

process of clay sculpting as a robotic planning problem; that is, given a user's intended goal, a robot takes a sequence of molding actions to create a sculpture matching the goal.

In robotics, clay sculpting is related to deformable object manipulation where the general goal is achieving a target 3D shape. Existing robotic sculpting approaches stemmed from such a 3D shape matching view tend to ignore important visual properties of sculpting, such as textures and shading due to lighting. For instance, subtle changes in textures can create drastic effect for human visual perception of sculptures, but it is hard to measure such an impact when using a 3D metric such as Chamfer Distance on sparse point clouds. To capture such visual guidance as that caused by lighting, we propose a robotic sculpting approach that plans in both a 3D and visually-aligned representations.

Although many robotic clay sculpting methods focus on additive or subtractive methods which do not model the dynamics or assume rigidity of the medium [118, 119, 120, 121, 122], there are works that model and embrace the softness and challenges of deformable materials. Some of these works plan using learning from demonstrations [11, 34], but these require retraining policies and recollecting demonstrations for each new goal and starting state. Avoiding this issue, other works model the clay dynamics and plan using policies or Model Predictive Control (MPC) [12, 123, 124].

When looking at clay, people do not only see the 3D aspects of the state, they perceive the way light hits the surface and the textures on the clay [125, 126]. Previous

work in robotic sculpting model dynamics and plan with sparse (∼300) point clouds [12, 34] which lack a direct visual interpretation and do not capture important features such as texture. We investigate whether a robot can plan to sculpt in both 3D and visually-aligned representations. We use dense depth maps (512×512) as a 3D representation and the spatial gradient of the depth map as a visually-aligned representation. Spatial gradients capture low-level changes to 3D surfaces and are essential representations used in rendering 3D objects into RGB images, as they are used to estimate the way a given light source interacts and reflects off the surface of the material. Therefore, we consider spatial gradients as a visually-aligned representation, with the assumption that two similar spatial gradients will have similar visual properties.

Clay sculpting is a long-horizon task which may require moving large amounts of material across the working area. Previous works in deformable object manipulation use pinches as actions using a parallel jaw gripper with 3D printed end-effectors [12, 34, 123, 124] which are designed for efficient creation of shapes like alphabet letters. Having two points of contact makes it challenging to make small details, and the pinching motion is not well-suited for moving material across a working area. In this work, we design our actions as simple pushes using a single end-effector to make simple, controlled deformations and better support long-horizon sculpting tasks.

We present a full system for robotic sculpting with visual planning. We represent actions as linear pokes along the surface of the material. We devise a self-supervised data generation scheme and train a dynamics model to predict deformations to depth maps of the material. The dynamics model predictions are differentiably converted to visual representations for planning which is performed using MPC to fit the predicted state to the target state. Comparisons are made both in 3D (e.g., chamfer distance) as well as visual (e.g., mean-squared error of spatial gradients). We show that this leads to sculptures that are not only accurate in 3D but visually similar to target states. Our contributions:

1. **A fully-integrated robotic sculpting system** capable of adapting to diverse materials and tools for performing long-horizon deformable manipulation.

2. **A deformable material dynamics model** that learns through limited self-generated actions and adapts to diverse materials and end-effectors (both soft and rigid).

3. The first robotic deformable object manipulation planning algorithm which **plans in both 3D and visually-aligned representations**.

7.2 Related Work

7.2.1 Deformable 3D Modeling

There have been exciting developments in computer-based 3D model generation and manipulation stemming from the rise of large models trained on vast datasets. Many works use 3D representations of Neural Rendering Fields [5], Gaussian Splatting, Meshes, or Point Clouds. These works introduce powerful ways to generate 3D models from images, text, or without condition. These methods provide very powerful tools, such as Score-Distillation Sampling to optimize the shape of a 3D model to fit a given text prompt [5], but they are not connected to real-world materials nor have relation to the actions and capabilities of a robot. There are existing material simulators such as PlasticineLab [127] which use methods such as the Material Point Method (MPM) to estimate the properties of real-world materials, such as clay. Despite improvements in these simulations, previous work has found that simulation methods such as MPM may not perform as well as data-driven approaches (e.g., graph neural networks) [12, 123].

7.2.2 Robotic Sculpting

Many robotic sculpting works utilize variants of subtractive or additive actions. Robots have used hot wires to cut through foam [119], loop tools to remove slices of clay [121, 122, 128], and chisels to carve wood [120]. While these subtractive methods are highly successful at recreating target 3D goals, this success in part comes from the assumption that the materials behave non-deformably. Even subtractive works using clay assume that the tool's path through the clay cleanly removes pieces of clay without creating deformations [121, 122, 128]. This assumption may work well for hard clay or styrofoam with sharp or hot tools, but will not hold up for very soft materials such as dough or sand.

Prior work has also created sculptures using robotics in an additive manner. These approaches are similar to 3D printing, in which materials are extruded and layered [118, 129]. These works do assume deformable properties of the materials, but they heavily engineer the systems to account for this (e.g., extruding very small amounts of clay at a time). These approaches are inherently additive, meaning that they cannot plan to change the existing state of the materials.

7.2.3 Robotic Deformable Manipulation

Rather than assuming rigidity, there are robotic works that can plan with the deformability of materials. Some works embrace deformability but do not explicitly model dynamics using learning from demonstration [11, 34] or large language models [130, 131] for planning. Other works plan with dynamics models which have been implemented using existing simulators, such as the material point method [12, 127], or

by training neural networks for a data-driven dynamics modeling [12, 123, 124]. With more diverse materials, works plan with deformable bags [132] and cloth [133, 134]. Although these dynamics models have decreased the Sim2Real gap, they represent clay state as sparse point clouds which do not capture low-level, visual details of clay such as textures.

Sculpting is long-horizon task; however, most deformable object manipulation works are focused on sculpting simple shapes (e.g., alphabetical letters) from a top-down view, which can be achieved in less than 10 actions. Prior works use parallel jaw grippers with custom end-effectors and represent actions as pinches [12, 123, 124]. While the pinch action is highly compatible with the alphabetical letter creation task, because of the multiple points of contact and large changes each action makes, it is not suited for long-horizon, fine-grained sculpting such as making relief sculptures with hundreds of actions. In RoPotter [11], a robot wielding a single end-effector was able to make pots, a task that traditionally uses at least two points of contact. In this work, we also use pushes with a single end-effector as actions to more closely align with sculpting tasks more generally.

7.3 Approach

7.3.1 Hardware Setup

Displayed in Figure 7.2, we use the UFactory XArm 850 robot with various, custom end-effectors. The sculpting surface is 12×12 inches with Aruco tags on the corners. Suspended directly above the sculpting surface is a Zivid One$^+$ structured light RGB+Depth sensor.

7.3.2 Action Representation

Robot actions are parameterized as linear pushes with a starting coordinate (x, y); a direction θ; a travel length l; and a depth component z, as depicted in Fig. 7.4. The action depth is with respect to the surface depth, starting at just lightly touching the surface then pushing z millimeters into the surface by the termination of the trajectory. z and l have maximum values that are set as hyper-parameters.

7.3.3 End-Effectors

We tested our system with multiple end-effector sizes, shapes, and levels of compliance (Fig. 7.3). We chose these end-effectors for their diversity. Our dynamics modeling approach is data-driven, and therefore should adapt to many different shapes and materials of end-effectors without needing prior information, such as 3D model. We also tested a gripper with two custom 3D-printed end-effectors as a baseline

Figure 7.3: End-Effectors. - We test our robotic sculpting system with a variety of single end-effectors of various shapes and levels of compliance and compare to a gripper which is conventional in prior work.

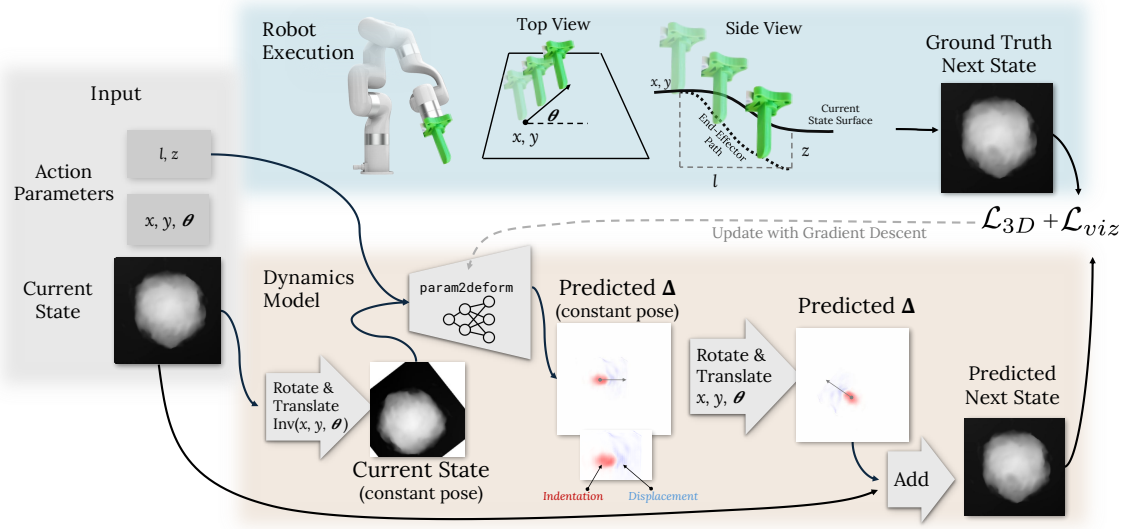

Figure 7.4: Dynamics Model. Given the action parameters and current state, our robot can follow trajectories to make deformations along the surface of the material. We model these deformations by training a neural network, `param2deform`, to predict the changes in state at a constant pose.

comparable to prior works. When using the gripper, we replace the l action parameter with the distance by which the gripper should be closed.

7.3.4 Dynamics Model

The goal of the dynamics model is to predict the change in the material's state given the current state and the action parameters. Our dynamics model is similar to the robot painting system FRIDA [107] with notable differences: (1) instead of brush strokes our actions are linear pushes along the material surface, (2) rather than predicting RGB brush stroke appearances our model predict changes to depth maps,

and (3) FRIDA assumes that brush stroke actions are independent of the current state, which is not true in sculpting, so our dynamics model includes the current state when predicting changes.

Fig. 7.4 shows a visual depiction of our dynamics model. At the core of the model is a neural network, `param2deform`, which predicts the change in depth. `param2deform` is comprised of 3 multilayer networks. The first takes the two shape parameter scalars, l and z, feeds them through 3 linear layers, resizes them to a 2D matrix, then feeds them through 3 convolutional layers. The current state is concatenated with its spatial gradient, then encoded using 3 convolutional layers. Finally, the current state features, current state, and encoding from the shape parameters are concatenated then fed through 5 convolutional layers to predict the final change in state. This network architecture was chosen for its simplicity and hyper-parameters (e.g., number of convolutional layers and hidden layer sizes) were tuned by hand.

To reduce the training data needed to learn the model, `param2deform` predicts all deformations in a constant pose, meaning the start of the action is always at the same point and the action moves from left to right. This predicted deformation is then translated into the desired position (incorporating the x, y, and θ parameters) using perspective warps which do not require training data to perform and are fully differentiable.

Dynamics Model Objectives

We sample random actions and capture scans of the depth of the materials before (S_t) and after (S_{t+1}) forming training data for `param2deform`. Our dynamics model, f, is designed to be both accurate in 3D and visual representations. We optimize `param2deform` with two different objectives to achieve these goals. Our 3D loss function, $\mathcal{L}_{3D}$ (Eq. 7.1), is the mean-squared error between the actual depth map after the action and the dynamics model prediction. To capture the visual features, we form a visual loss function, $\mathcal{L}_{viz}$ (Eq. 7.2), which is the difference in spatial gradients of the actual depth map after the action and the dynamics model prediction.

$$\mathcal{L}_{3D} = ||S_{t+1} - f(S_t, a)|| \tag{7.1}$$

$$\mathcal{L}_{viz} = ||\nabla S_{t+1} - \nabla f(S_t, a)|| \tag{7.2}$$

We can also convert the depth maps into point clouds and compute more standard loss functions like Chamfer Distance (CD) and Earth-Mover's Distance (EMD). Since the depth maps are high-resolution (512×512), we must down-sample our point clouds for computational purposes. We use voxel-grid down-sampling and the computations of CD and EMD from [12].

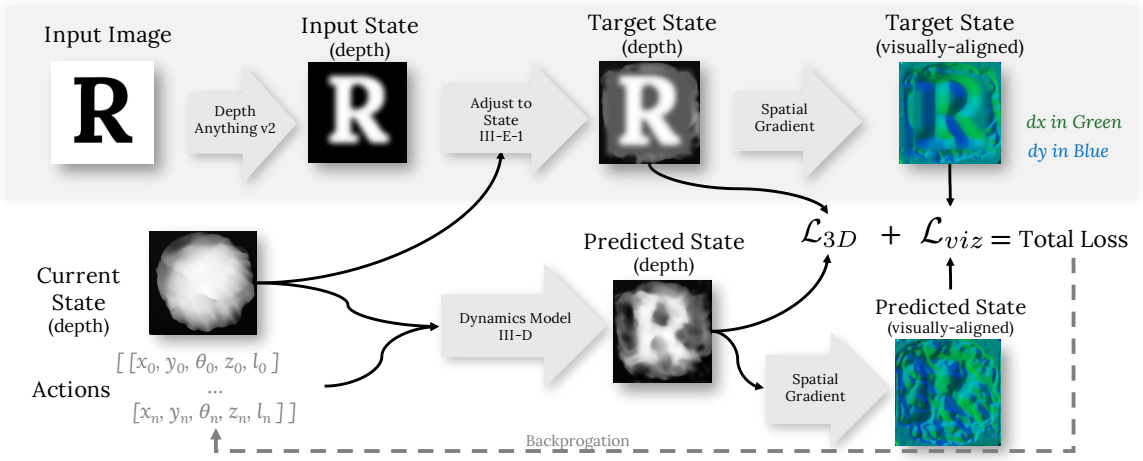

Figure 7.5: Planning. (Above) An image is specified by a user and is then converted to depth. The depth map is altered to make it more feasible for the robot to create based on the current state of the material forming a target state. (Below) Our planning algorithm optimizes a set of randomly initialized actions such that the dynamics model predicted state is both accurate in 3D and visual representations compared to the target state.

7.3.5 Planning

The goal of our planning algorithm is to recreate a given 3D model in both a 3D and visual representation. The number of actions, starting state, and robot end-effector choice are given. We employ a combination of random-sampling with Gradient Descent for optimizing the action parameters to achieve this goal.

Goal Creation and Processing

Our planning algorithm requires a dense, depth map as the target state representation, which can be given directly as a depth map from a 3D model or can be extracted from a given image input by using the pre-trained image-to-depth model, DepthAnythingV2 [135].

The input depth map is not calibrated to the given starting state, so, for example, there may not be enough clay in the current state to recreate the target depth map. We adjust the input target depth map (S_t) with the current state depth map (S_0) with an optimization of the surface. Shown in Eq. 7.3, we scale the target depth map by α and β. Because it is difficult for our robot to work near the edges of the working area, we ensure that the edges of the final target depth, $\hat{S}$, are equal to the current state using boolean map, M, which isolates the outer 10% of the working area. Our optimization function, Eq. 7.4, optimizes scalars α and β such that the target depth map, $\hat{S}$, has (1) the same amount of material as the current state, (2) does not have depth values larger than the table surface, d_{max}, and (3) has the most definition

(defined as a large α value). These three terms are weighted ($w0$, $w1$, and $w2$) by hand from experimenting with a few test cases.

$$\hat{S} = S_0[M] + (\alpha S_t + \beta)[1 - M] \tag{7.3}$$

$$\min_{\alpha,\beta} w_0(\Sigma\hat{S} - \Sigma S_0) + w_1\Sigma\hat{S}[\hat{S} > d_{max}] - w_2\alpha \tag{7.4}$$

Planning Objectives

Similar to our dynamics model objectives, our planning objective is to recreate both the 3D shape and visual attributes of a given shape model. We optimize the action parameters, $\mathbf{a}$, to achieve these objectives (Eq. 7.7). We compare the 3D shape of the target shape, S', and our dynamics model prediction, $f(S_0, \mathbf{a})$, using mean-squared error matching pixels of depth maps.

$$\mathcal{L}_{3D} = ||S' - f(S_0, \mathbf{a})|| \tag{7.5}$$

$$\mathcal{L}_{viz} = ||\nabla S' - \nabla f(S_0, \mathbf{a})|| \tag{7.6}$$

$$\min_{\mathbf{a}} [w_{3D}\mathcal{L}_{3D} + w_{viz}\mathcal{L}_{viz}] \tag{7.7}$$

Planning Algorithm

Our planning algorithm is a simple variant of MPC. A given number of actions are initialized using greedy sampling. Actions are initialized one-by-one picking the action that decreases the loss the most over a number of trials. These initialized actions can then be optimized using gradient descent or cross-entropy method to decrease the loss values. While the initialization is greedy, this optimization stage helps promote long-horizon planning since all actions are influential in the objective. This forms an initial plan. A small number of actions are performed, then the robot pauses to update and optimize the remaining plan, to adjust for differences in the dynamics model prediction and reality. This process is repeated until all of the actions in the plan are performed.

7.4 Results

7.4.1 Dynamics Model

Qualitative & Quantitative Results

To test the accuracy of our dynamics model and investigate the effect of our visual loss term, we train our dynamics model with various materials using the

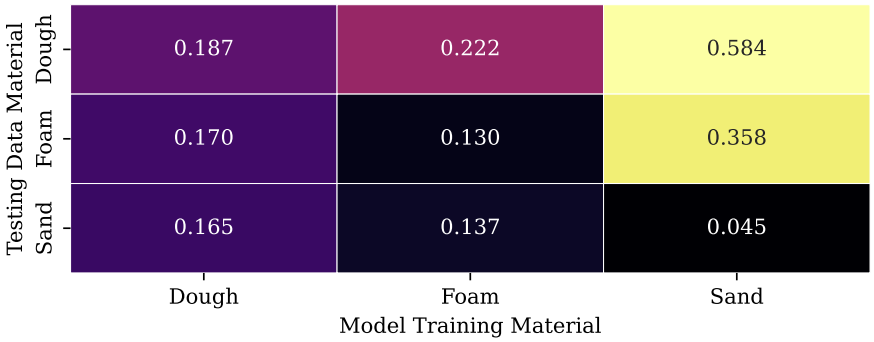

Figure 7.6: Out-of-Distribution Dynamics Modeling. We train our dynamics model on one material and test on another. Reported above are Sim2Real gap values (lower is better) computed as the MSE between predicted and true depth maps (Eq. 7.1).

same end-effector and show the performance on various metrics on a held-out set of deformations in Table 7.1. Generally, the addition of the visual loss term, $\mathcal{L}_{viz}$, increased performance not only on the visual evaluation metric but also on the 3D metrics.

For qualitative investigation of our dynamics model, we displayed sample predicted deformations in Fig. 7.7. Overall, our model is able to capture the complex deformations in various materials in the local region but fails to predict small deformations far away from the contact. Between the tested materials, foam hosted highly complex deformations while sand had unpredictable deformations that depended greatly on the gradient of the surface (sand rolling down a hill).

Generalization Across Materials

In Fig. 7.6, we evaluated our dynamics model trained on data from one material and then tested data from another material. We found that some materials lead to good generalization, as a model trained on foam and tested on dough performed only slightly worse than that trained on dough itself. However, training on sand led to very poor generalization to other materials. Overall, these results show that deformable materials are nuanced and cannot be modeled by a single set of parameters.

Dynamics Model Sample Efficiency

We trained our dynamics model with varying numbers of training samples and displayed the results in Fig. 7.8. Our model performs well with only roughly 100 samples, but performance increases with more samples.

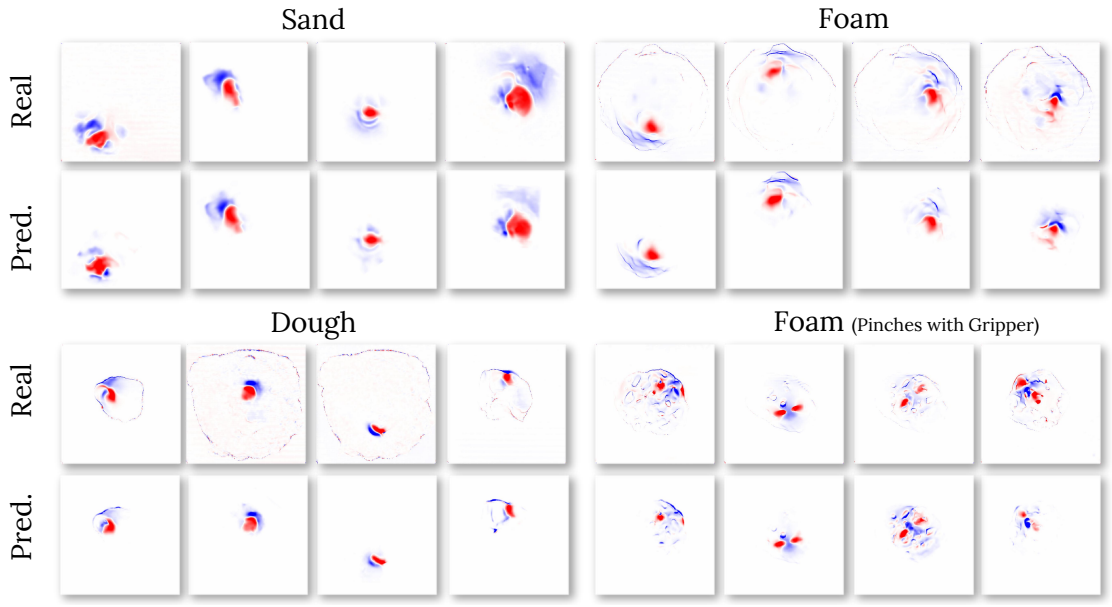

Figure 7.7: Qualitative Dynamics Model Results. The top rows show real deformations made into various materials by our robot. Our dynamics model predictions given the current state and action parameters are shown below the real deformations.

Table 7.1: Visual Dynamics Modeling. - Dynamics model performance on a held out set of deformations with various materials while ablating the training objectives. Lower is better for all metrics.

	Objective(s)	$\mathcal{L}_{3D}$	$\mathcal{L}_{viz}$	CD	EMD
Foam	$\mathcal{L}_{3D}$	0.138	0.025	0.26	0.16
	$\mathcal{L}_{3D} + \mathcal{L}_{viz}$	**0.130**	**0.024**	**0.22**	**0.15**
Sand	$\mathcal{L}_{3D}$	**0.043**	0.012	**0.40**	**0.22**
	$\mathcal{L}_{3D} + \mathcal{L}_{viz}$	0.047	**0.011**	**0.40**	**0.22**
Dough	$\mathcal{L}_{3D}$	0.190	0.029	0.45	0.31
	$\mathcal{L}_{3D} + \mathcal{L}_{viz}$	**0.187**	**0.028**	**0.41**	**0.30**
Foam Pinch	$\mathcal{L}_{3D} + \mathcal{L}_{viz}$	0.624	0.043	0.50	0.30

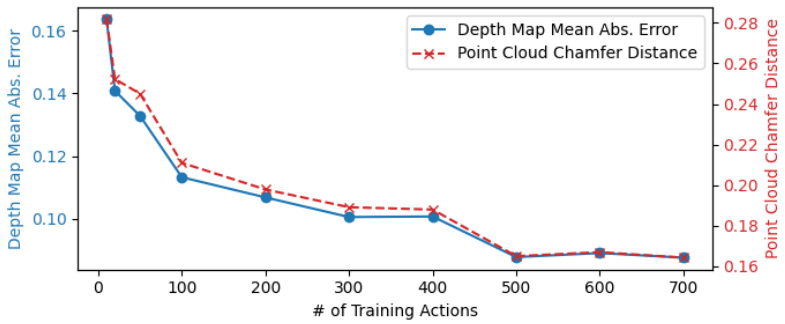

Figure 7.8: Dynamics Model Sample Efficiency - Our dynamics model is able to learn an accurate transition model with as few as 100 actions.

7.4.2 Planning Results

Long-Horizon Planning

To test our method's ability to handle long-horizon tasks, we create a series of goals of alphabetic letters in serif font. The robot first sculpted an "A" from a starting state, then morphed it into a "B" and so on. In Fig. 7.1, we report results to "F", showing the robot's capability to plan over hundreds of actions that vastly altered the material's state.

Pinching versus Pushing

We compared our single end-effector pushing actions with a pinching action using a gripper that is analogous to other deformable manipulation works [12, 34, 123, 124]. In Table 7.1, we reported the results of dynamics modeling which were worse for the pinching actions, indicating that pinches produced less predictable deformations than our pushing actions. This was supported by qualitative results in Fig. 7.7 which showed that the pinches were complex and not modeled as accurately. In Fig. 7.11, both visual and 3D losses decreased as many actions were taken using the single end-effector pushes, however, only the 3D losses improved with the gripper pinches. We found that the pinches produced choppy, messy sculptures over the course of many actions.

Comparison with Other Dough Works

In Fig. 7.9, we compared our approach to existing deformable object manipulation works. Since we were unable to replicate results from these works, the images of the results were taken from the papers along with the metric results. We attempted to use similar starting and goal states to the compared works. Although our approach was designed for larger, long-horizon sculptures, this comparison served as preliminary evidence that our approach is comparable on the task of making simple, small shapes

to existing approaches [12, 34, 123]. However, because of the lack of control in this experiment, we are unable to draw broader conclusions about the differences in results.

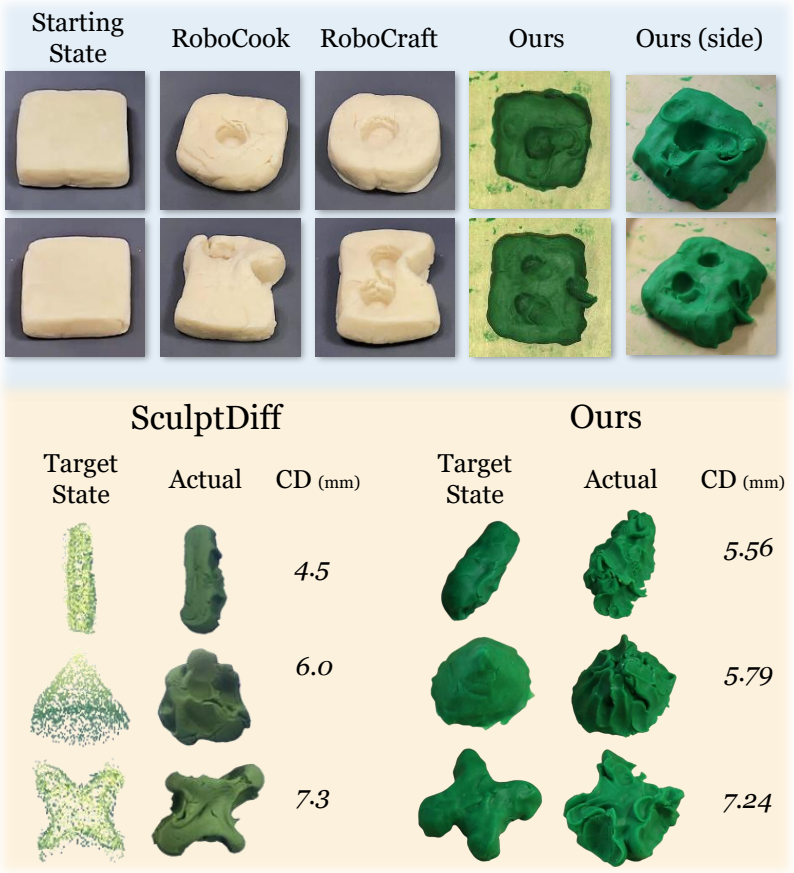

Figure 7.9: Simple Shape Results - Our method has similar results with simple shapes, such as letters and pyramids, to other play-doh manipulation works, RoboCraft [12], RoboCook [123], and SculptDiff [34].

Goal Creation and Processing

In Fig. 7.10, we show two examples of the processing steps when receiving an input modality to be used as a target shape. In the upper example, an RGB image is generated by an image generator, then the depth is extracted using DepthAnythingV2 [135]. Using the current state, this extracted depth is adjusted according to the optimization described in Sec. 7.3.5. The second example in Fig. 7.10, shows an example where a 3D model downloaded from the Internet is converted to depth, then adjusted according to Sec. 7.3.5.

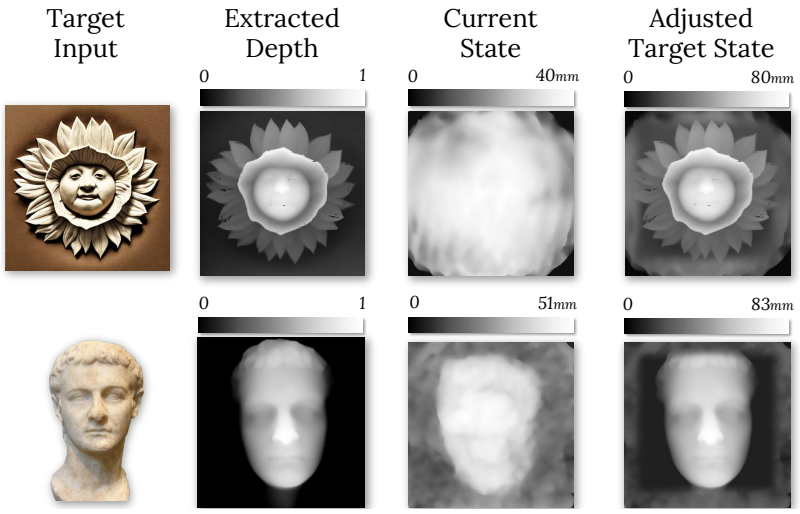

Figure 7.10: Goal Creation. Target depth maps are adjusted so that they are more feasible for the robot to recreate. Details in Sec. 7.3.5

3D-Visio Planning

To observe our approach's ability to improve 3D and visual accuracy over many actions, we plotted the losses as actions were taken as the robot sculpted a large "X" relief sculpture in Fig. 7.11. Our approach (with a single end-effector) is able to decrease both the visual and 3D losses over many actions, though, it is worth noting that the visual loss did not decrease steadily and leveled off before the 3D loss.

To investigate the effect of 3D and visual losses, we performed an ablation study of the losses in the planning objective and reported the results in Fig. 7.12. We created a simple case where the robot's objective was to smooth out a thin line pinched into the material. This example showed an extreme change in visual representation, whereas the change in 3D was not as extreme because the line was very thin. When planning with a point cloud representation and chamfer distance, the robot did not smooth out the line well, whereas when planning with visual loss, most of the robot's actions worked to smooth the line. We attempted a more complex example in Fig. 7.12, where the robot's goal was to create a ripple in the sand. The effect of visual planning did not appear very strong here, even though quantitatively the visual guidance was supportive. We hypothesize that this was because it is very challenging to make smooth surfaces on sand, as even the lightest touch with the end-effector tends to make a strong visual indentation.

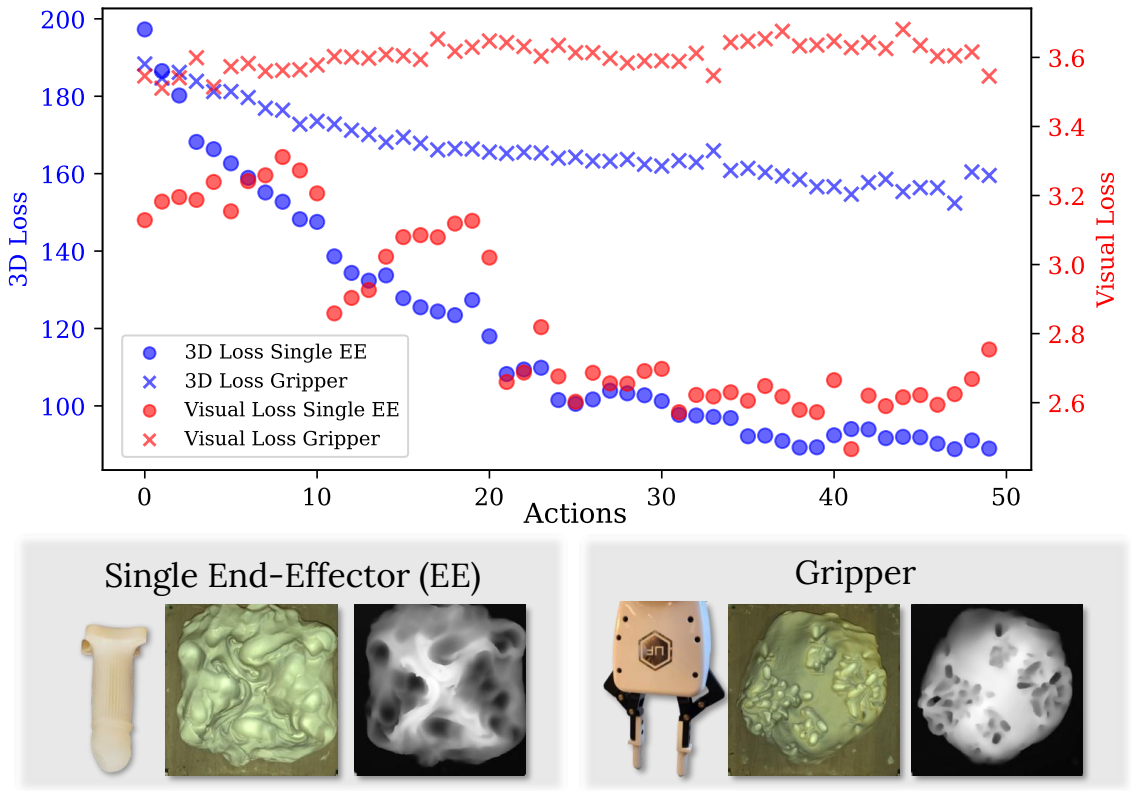

Figure 7.11: Visual and 3D losses during long-horizon sculpting. The losses were plotted after each of 50 actions taken by the robot using a single end-effector with our pushing actions and compare to a gripper using pinch actions analagous to prior works [12, 34, 123, 124]. Below, we show samples of photographs and depth scans of the material after the actions were taken.

7.5 Discussions

7.5.1 Limitations

A large technical limitation of our approach stems from our RGB+D sensor which is very expensive and fixed in a static location leading to a single perspective 3D view. It would be possible to plan and model the dynamics of multiple perspectives to get a truly 3D sculpting method if the sensor could move to additional positions. Another limitation of our approach is the simple action parameters that are executed only from the top down. For a more expressive approach with more 3D capabilities, more complex actions could be designed.

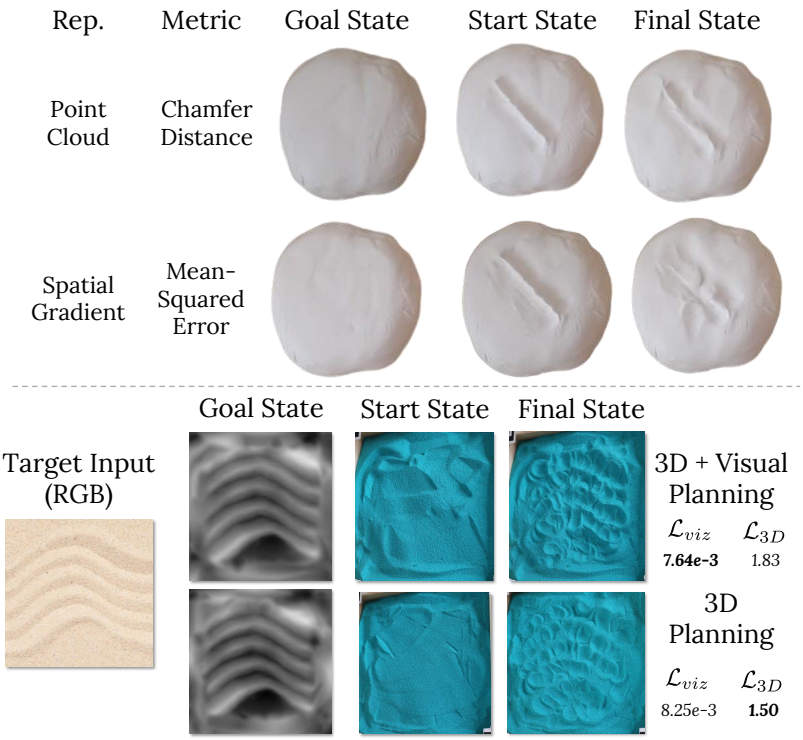

Figure 7.12: Sculpting in a Visual Representation. (Above) We isolate planning in 3D (minimizing Chamfer Distance) and visual (minimizing mean-squared error of spatial gradients) representations. In both conditions, the robot performed 10 actions to smooth out a line pinched in the clay. Planning in a visual space creates plans that properly align with the task. (Below) When planning with more complex goals with sensitive materials (sand), the effect of visual guidance was not as apparent.

7.5.2 The Sensitivity of Visual Guidance to Noise

In multiple experiments including dynamics modeling (Table 7.1), long-horizon sculpting (Fig. 7.11), and smoothing surfaces (Fig. 7.12), we observed that our robot is successfully able to use visual guidance, in addition to 3D guidance. However, this visual guidance sometimes had little or no effect, as was seen with the sand example in Fig. 7.12. We hypothesize that this is a result of the visually-aligned representation being highly sensitive to noise and less unpredictable than 3D representations. In Fig. 7.13, an action applied to a relatively flat surface had a simple change in depth, with a large indentation surrounded by some displaced material. However, in the ray-traced visual representation, this change is more complex.

We hypothesize that the Sim2Real gap has a more drastic effect on visually-aligned representations compared to 3D representations. We can simulate a Sim2Real gap by adding Gaussian noise to the action parameters of a planned, simulated sculpture. In Fig. 7.14, the losses were plotted as the amount of noise added to the actions increased. We also plotted the loss if zero actions were taken as horizontal, dashed

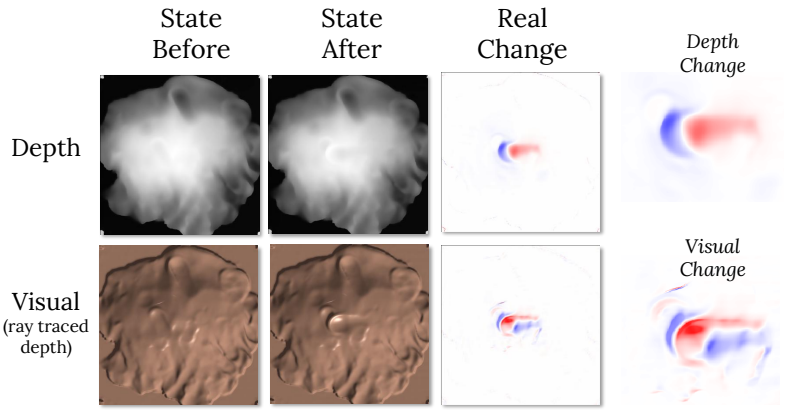

Figure 7.13: Sensitivity of Visual Representations. Depth maps are shown before and after an action is taken along with ray traced conversions of each. The changes in depth appear less complex than the change in ray traced images (averaged over RGB channels).

lines. As expected, both the 3D and visual losses increased with additional added noise. However, the visual loss reached a point where the robot is better off not taking action with a smaller amount of noise than with 3D loss.

In Fig. 7.14, we measured the amount of noise as the MSE between the predicted and the noised actions which is the same calculation as $\mathcal{L}_{3D}$ in Table 7.1. The point where the robot is better off not performing any actions with respect to visual loss is roughly 0.2 in Fig. 7.14. As seen in Table 7.1, our Sim2Real gap is between 0.05 and 0.19, depending on the material, indicating that our method is barely able to perform visual planning without the Sim2Real gap being too high. This experiment supports our hypothesis that visual planning is very challenging due to the Sim2Real gap, but it is just one experiment and more evidence is needed to make broader conclusions.

7.6 Conclusions

7.6.1 Generalization from Painting to Sculpting

This chapter of the thesis served to generalize our approach beyond 2D painting settings. We showed that our approach to Generative Robotics can generalize from 2D states to 3D by representing clay materials as depth maps. Tested on clay, dough, and sand, our approach was able to learn from self-generated actions the dynamics of these materials with respect to actions. Sculpting action representations are similar to painting actions but extended for a vertical component. With minimal adjustments, our approach scaled from painting to sculpting.

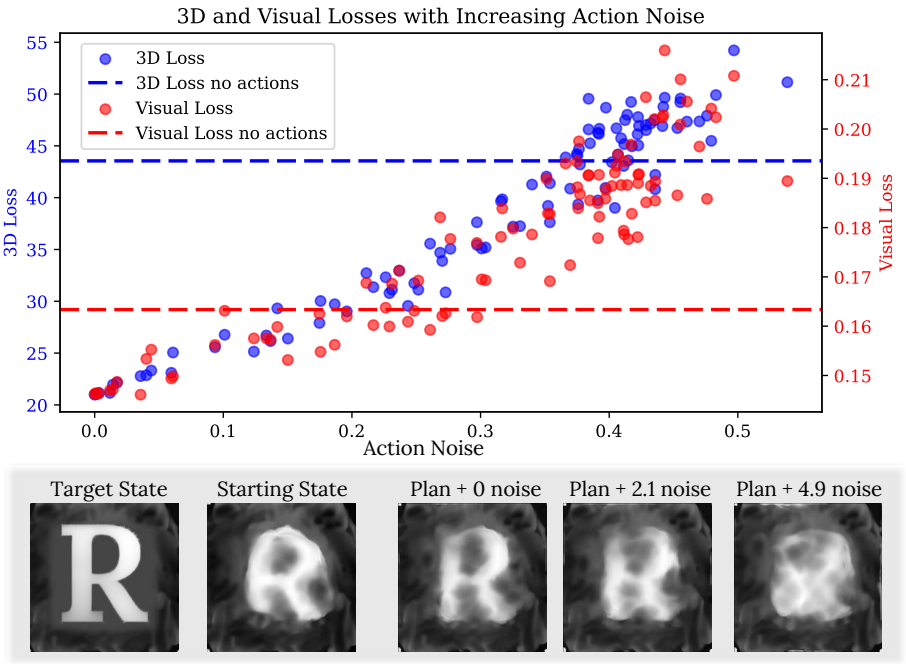

Figure 7.14: Noise versus Visual and 3D Accuracy. Above, we plot the visual and 3D losses as more Gaussian noise is added to planned action parameters, simulating real-world noise. Samples of depth maps of plans with increasing noise added are shown below the plot.

7.6.2 Challenges in Robot Sculpting

While our sculpting approach improved upon existing approaches in many respects, such as long-horizon planning and high-res dynamics modeling, the results are still very poor compared to that of even an amateur sculptor. This is in stark comparison to our painting results which arguably look similar to that of an artist with some skill. Our results point to a few different reasons as to why our sculpting results are so poor compared to our painting results.

Isolation of Actions. A brush stroke made to a dry canvas makes a highly-localized modification to the state of the canvas, however, a small push on the surface of clay can change the state globally in sculpting as seen in Figure 7.7. Because the actions have such extreme effects, it makes it challenging to perform long-horizon planning, especially if the dynamics model is not perfect. One way to improve upon this would be to add some methods for planning with uncertainty to try to take safer actions towards a goal [136].

Action Limitations. While our pushing actions improved long-horizon planning compared to the pinches, as seen in Figure 7.11, these actions are still very limited in their ability to make complex sculptures. A human sculptor may use many tools or complicated, dexterous maneuvers to make desired shapes. Our approach could be improved with more types of actions including some that are subtractive or additive.

End-Effectors. The choice of end-effector has a great effect in what is possible to create with the robot. Larger end-effectors obviously have trouble making small details. Compliance turned out to be an important factor to consider. Softer end-effectors were able to create smoother surfaces, while more rigid end-effectors created ridges and texture with even the lightest touches. In future work, exploring more compliant tools and perhaps tools with variable stiffness could improve sculpting accuracy.

Material Unpredictability - Changes to clay, sand, dough, and generally all deformable materials are very challenging to predict. Our dynamics model is far from perfect, but its also not clear what is the maximum performance it could get. One way to work with unpredictable materials is to improve sensing. Our planning is closed-loop, but our actions are conducted open-loop. If we were to sense and respond during action execution we could try to control uncertain factors. For example, contact estimation could be more precise with a force-torque sensor. Using tactile sensing like GelSight could also help improve perception of the clay state [137].

7.6.3 Improvements to Robot Sculpting

Despite the shortcomings described in the previous subsection, our method was able to improve upon the state of the art in multiple challenges of sculpting. Our method was able to plan for hundreds of actions (Figure 7.1) compared to prior work which plans with less than a dozen showing how are robot can robustly plan for long-horizon tasks. Our state representation of depth maps of size 512×512 is far richer than prior work that generally rely on sparse point clouds. Our insight that pushed would be a more amenable action representation than pushes for long-horizon sculpting proved to be true through our experiments. Lastly, our dynamics model showed promise for improving state of the art prediction of deformations given actions.

V

Conclusions

8 Conclusions

8.1 Discussions

This thesis touches on many areas of robotics and creativity. In this section we discuss interesting questions that arrise when combining AI, robotics, and art.

What is the difference between a printer and a robot painter?

A 2D laser printing machine can recreate almost any image given to it using specialized mechanical parts and ink. While a human painter is also creating images, their occupation is quite different from a printer and a robot painter follows suit. We identified two major differences between a painter and printer in this thesis. (1) The tools and materials should affect what is created in painting. A robot working with a single black Sharpie marker will necessarily create different content from a robot with diverse oil paints. This is in contrast to a printer which can create anything and does not change the content based on its mechanical parts or ink. (2) Painting is a process. The goal may change drastically throughout the painting process as the artist makes iterations, as opposed to a printer which simple takes the image and recreates it without iteration.

These two differences between a painter and printer drive the technical challenges of robotics in this thesis. The robot has to have an accurate understanding of its real-world materials, tools, and abilities in order to let them influence what is created. In robotics, this is challenging because art materials, such as paint and clay, are very difficult to model in simulation. The robot also has to have flexible, high-level goals and allow for changes through the process to support the iterative nature of art. This is notoriously challenging in robotics because it requires high-level intelligence.

What does it mean for a robot to collaborate with someone when creating?

Support can come in many forms during a creative process. For example, someone may help proofread a paper you are writing or maybe they just provide you with some

snacks as you write into the night during a deadline. Both are forms of support but look very different. There is no one size-fits-all solution to supporting creative acts. In this thesis, we explored collaboration as a form of support in painting. We also observed during live demonstrations that the robot was able to support artists by inspiring them and motivating them.

During demonstrations, participants would often not want to start a drawing by themselves. Instead, they were more comfortable typing in a text prompt and having the robot start it. After this, they felt more comfortable to start drawing. While more proper experiments need to be run to find more general conclusions, we theorize that robots can help motivate people to engage in art more. During these same demonstrations, the robot would often draw something unexpected. While this could potentially be frustrating if the participant wanted to draw something very specific, when people had open minds, it helped inspire them to draw something new and exciting. There are many possibilities for how a robot can support the creative process, and we look forward to investigating these in future work.

How can a robot respond to changes in a creative process?

Generative Robotics tasks are process driven in nature, meaning that they can be iterative and goals can change throughout the task. More generally in robotics, there can be goal shifts and moving targets in many tasks. In this work, our robotic system can respond to two different types of changes in the process. High-level goals may change throughout the process. For example, a person may want to draw a simple bucolic scene at first, but may want to change to a more sci-fi scene towards the end of the painting, as scene in Figure 6.2. CoFRIDA is capable of responding to this because our system self-generated training data to support this type of goal change.

Even though the performance of our dynamics model is good, it is not perfect. There are Sim2Real gap errors that can pile up over a long-horizon painting task. These are examples of low-level, stochastic changes that the robot needs to respond to. In Figure 4.11, our robot planner responded to this change, adapting its plan to account for it. Our fully-integrated system is able to perceive and re-plan accordingly.

How to make concrete plans for high-level goals?

A question arose while working on robot drawing: How can the robot draw from a source color photograph if it only has a black Sharpie marker? The question gets at ideas around robot goal aligment. In this case, the robot isn't supposed to recreate the image *pixel-for-pixel*. Instead, its goal should be to recreate *the content* of the image. This inspired our approach for planning in a semantically aligned representation. With semantic planning, the robot is able to draw portraits of people from color photographs of them, as seen in Figure 4.15. In the painting space, we observed that people found the paintings using semantic planning to look more like the source photograph than

the ones planned with mean squared error even though they were less similar in a pixel space (Figure 4.14).

In many manufacturing tasks, the goal is achievable by the robot because the robot capabilities were exceptional and/or the goal was designed to be achievable by the robot. In this thesis, we work on problems where the robot cannot create the goal state exactly. For example, a brush is too large to make small details like tree leaves (Figure 4.13). To help robots be more general purpose and work with diverse materials, they need to be able to make things that align with the user's goals. Our planning in a semantic representation method proved helpful with increasing this alignment in heavily constrained settings.

Can a robot teach itself about how its tools and materials interact?

Models of the world and how the robot can take actions in it can be helpful for planning. One way to do this is through a simulation grounded in physics definitions, however, if the simulation is not accurate, this can lead to large Sim2Real gaps. This can be very apparent when a robot is using specialized, unique tools like a paint brush or sculpting knife. Additionally, materials, such as deformable clays or paints, are highly unpredictable. Many works will tune a few parameters of a simulation to try to decrease this Sim2Real gap either by hand or with data. In our work, we train a neural network from scratch using actions that were sampled randomly as training data and found that it was more accurate than the hand tuned approach (Figure 4.10). While this is not the first case of using neural networks as dynamics models, ours learns a very complicated relationship (action parameters to brush stroke appearance) with very few training samples needed because of our translation and rotation invariant dynamics model. This generalizes to the 3D space in our sculpting work as well.

Our robot self-generates actions to train the dynamics model. In the painting setting this works well because there is little variation in the current state that would affect the appearance of the actions. In sculpting, however, the current state has a huge influence on the action's appearance. While our random sampling scheme worked well for sculpting, a more intelligent sampler could lead to bigger improvements in the dynamics modeling as it could find more unique cases for training.

To mimic a human or to find a new way?

For various reasons (e.g., embodiment differences or efficiency), a robot may perform a task in a very different from a human. For example, a printer machine may create an image working from top to bottom and left to right. This would be very challenging for a human painter to do, instead, they may sketch with pencil, block in large areas with a brush, then finally paint small details throughout. In this thesis, the approach that the robot takes for sculpting and painting is shaped to some extent by the definitions of its creators, but in large part the robot figures out how to paint on its own. We defined the action representation leading to a set of possible brush

stroke shapes as well as restrictions such as the number of paint colors. But the order of the brush strokes and where to place them are all determined by the robot based on its objective.

The final paintings may look similar to those made by humans, but the intermediate stages look peculiar and strange. There are many works that try to model a human painting process [138, 139]. Like no two people paint with the exact same process, there is no need for a robot to perform similarly to a human. It only depends on the needs of the user of the system and the desired effect.

What is success for creative tasks?

In many domains, a simple and intuitive definition of success can be computed for internal or external evaluation. For example, in robot navigation, the distance between the robot's end position and goal position can be computed for understanding success. But in creative acts, such as painting or sculpting, it is not as straightforward. There is not a function or calculation that can determine if a painting that a robot made is successful. Instead, in this thesis we create proxy goals that are designed to align with the intentions of a human user. In CoFRIDA, a user wants the robot to draw according to a text prompt and what is currently on the state. Our proxy goal in this case, was a latent diffusion loss which guided the fine-tuning of a pretrained image generator to generate collaborative additions to an existing painting. In sculpting, the definition of success was the mean squared error between visual representations of the the 3D state. These proxy goals allow a robot to optimize and plan, while being aligned enough with a human user's intentions and goals.

Can Generative Robots behave ethically?

As AI and robotics enter creative spaces at scale, huge questions about ethics have arose. This topic is complex but important so we will comment briefly on it here.

Most large models were trained on huge datasets scraped from the internet, such as the LAION [24] dataset. While most images in this dataset are innocuous, there exists a significant amount of biased and terrible data [117]. The problems with the data can carry over into the predicted images. In our prior work, we were able to successfully mitigate some of the cultural biases of this data, but these systems could still be offensive [140, 141]. Besides bias, some of the training data was collected without the concent of the authors. This allows models to replicate styles which could harm the reputation and well being of the original artist. Issues like this need to be solved, however, to our knowledge there does not exist a solution beyond forbidding the usage of the models.

Can a robot be creative?

Although creativity is highly subjective and hard to define, there are generally two components of creativity: novelty and value. Novelty simply means new which could be new to a single person or globally (P-creativty versus H-creativity according to Margaret Boden [142]). Value can mean many things, such as monetary value, emotional provocation, colorful, or sentimental. For data-driven AI systems that learn using maximum likelihood estimation, they can generate valuable solutions easily since they are trained on many examples. For example, a generative adversarial network learns to generate images of cats well, and if someone is looking for images of cats, this is valuable. Where data-driven systems struggle to be creative is in novelty.

Data-driven systems can recognize outliers or out-of-distribution samples well, but it has trouble evaluating them since there is little or no data to support this. Creative samples are necessarily out-of-distribution since they are novel. Therefore, we argue that data-drive robots cannot be truly creative. However, this argument lacks concrete evidence and is based on strong assumptions. For example, Boden [142] defines combinational creativity as the combination of familiar ideas. By this definition of creativity, data-driven approaches appear to be very creative, as they can easily mashup concepts. However, for Boden's other types of creativity, exploratory and transformative creativity, where sampeles need to be completely new and out of distribution, data-driven approaches would be very unlikely to be creative.

Through our rhetorical argument, we state that a robot could not be creative on its own, but there is evidence that robots can support and augment human creativity [143, 144, 145]. Fostering good human-robot collaboration is a more promising area for future research than robot automated creativity.

8.2 Technical Merit

In this thesis, we show that self-supervised learning can adapt a model-predictive control approach to robotics for creative tasks. Specifically, we introduce three self-supervised learning techniques that use self-generated robot data to enable robots to perform tasks, such as painting and sculpting.

Real2Sim2Real Dynamics Model

Modeling the world and the way a robot interacts with it is one of the biggest challenges in robotics. In this thesis, we introduce a Real2Sim2Real, self-supervised learning approach to dynamics modeling. There are many previous works that tune parameters with real data to inform the model [49, 54]. Other works may train a model from scratch [12] using neural networks, but they often need to lower the dimensionality of the states or use simple models to avoid overfitting. In sculpting, 300 point clouds do not represent the visual qualities of the state. In our approach, we are able to keep dimensionality high (512×512 depth maps) while avoiding overfitting with our translation and rotation invariant dynamics modeling technique.

Our approach models tiny changes to the world with high accuracy as opposed to off-the-shelf simulators which have trouble modeling minute details. In this thesis, we showed that our dynamics model worked well for painting and generalized to sculpting. This shows promising results that the approach could be used to model the dynamics for many more tasks.

Planning in a Semantic Representation

We showed that performing robot planning in a semantic representation can increase goal alignment under constrained conditions. For example, planning in a visual feature representation allows the robot to draw portraits from color photographs even though the robot is constrained to only use a few black marker strokes (Figure 4.15). Planning in semantic representations uncovers huge technical challenges in robotics since these representations may be high-dimensional and complex. Because of these challenges, reinforcement learning has difficulty converging with such complex state representations. MPC provided a more feasible framework to converge; however, this was only possible with a differentiable dynamics model to enable gradient-based optimization for planning. This was another technical challenge, which was tackled in this thesis by using differentiable operations and neural networks in our dynamics model.

In conclusion, this thesis showed the importance of planning in a semantic representation and provided ingredients towards successfully generating plans: differentiable dynamics models used in MPC style planning.

Human-Robot Collaboration

In many creative tasks, the usage of robotics is better suited for collaboration rather than automation. Unfortunately, many of the largest datasets only support automation since the represent the final product. For example, there are millions of images of paintings, but very few samples of how the paintings were made or data that supports collaborative painting. To solve this, we created a self-supervised data creation technique to adapt pretrained models to be collaborative in this thesis. This method was highly successful in the collaborative painting domain, and we are hopeful it can extend to more collaborative robotics tasks, such as cooking.

8.3 Future Directions

The technical contributions of this work were developed for robot painting but proved to scale to robot sculpting. The contributions of this thesis can help in a number of Generative Robotics tasks ranging from creative industries to scientific discovery.

8.3.1 Mental Health Support

The World Health Organization has reported that practicing arts can help people experiencing mental illness, support people with neurodevelopmental and neurological disorders, support care-giving, and other well-being aspects [146]. Multiple prior works have developed art robots to support well-being [101, 102, 103, 104]. Robot co-creative assistants have been found to be motivating for people to engage in arts [97]. A robot art installation at a hospital was found to reduce anxiety and boredom in patients and patient visitors [105]. While there is no definitive evidence that Generative Robotics necessarily improves well-being, this prior works suggests that it is a valuable area to explore.

8.3.2 Personalized Touches in Mass-Manufacturing

Recent developments in systems that can generate images, videos, and text have raised concerns from professional artists in fear of their livelihoods being replaced [1]. One potential area where Generative Robotics could potentially induce more creative jobs is in mass-manufacturing. In mass-manufacturing, a small team of humans designs a product that is reproduce many times. For each product, there is less than one person who helped create it on average. In contrast, hand-made goods can often have a 1-to-1 ratio of creators to products. Mass-manufacturing is performed because cookie-cutter creation of products is inexpensive with current machines. Generative Robotics could reduce the amount of time it takes a human to create goods while maintaining their creative vision. With Generative Robotics at scale, you could have the scale in inexpensiveness of mass-manufacturing, but where each product has personalized touches by a human artist.

The technical challenges for personalize mass manufacturing lie primarily in how to use custom materials and assemble them such that it satisfies the designer. This thesis introduced a flexible dynamics model which can help the robot understand its capabilities and how to use various materials. We also showed how to use higher-level reasoning to fit the needs of a user with our semantic planning approach.

8.3.3 Upcycling & Recycling

Current mass-manufacturing practices rely on raw materials that were generated for the purpose of manufacturing, such as standard sized wood boards or raw plastics. Some materials, such as plastics, can be reclaimed and processed back into standardized materials for manufacturing [147], but other materials, such as textiles, are more challenging to reuse. Our Real2Sim2Real dynamics modeling approach could be used to improve existing clothing simulators to improve manipulation of textiles. Additionally, our self-supervised learning technique can ground foundation models in a realistic understanding of clothing such that the model makes predictions of clothing assemblies that the robot can actually perform.

8.3.4 Scientific Discovery

Creativity and discovery are very similar. Both have balanced notions of novelty and value. Whereas in creativity value may be defined as beauty or provocativeness, in discovery value may more closely be aligned with utility or efficiency. Because of the similarity of creativity and discovery, we argue that our approaches to creativity support in this thesis can also contribute to scientific discovery.

Self-driving laboratories are fully autonomous robotic systems that can create hypotheses and test them to attempt to discover new knowledge about the world. For example, UC Berkeley's A-Lab has discovered thousands of new materials autonomously [148]. However, some recent research has argued that most of these discovered compounds are not truly novel [149], and that a human expert in the loop could improve the abilities of these systems to understand novel circumstances [150]. These findings are consistent with our argument that AI cannot be creative since it struggles to find valuable solutions that are novel since there is no data to support these predictions.

The FRIDA framework for robotics could be adapted for scientific discovery. Instead of making paintings, the robot could be conducting experiments to create and test new materials. Rather than showing collaborative previews of paintings, the robot could suggest hypotheses that a human expert can decide if they are worth testing.

Serendipity is a cornerstone of scientific discovery. Penicillin was discovered by accident while Alexander Fleming was researching other ideas. But because of his human ingenuity and intuition, he caught this serendipitous result and followed up on it. Would AI be able to catch such a surprising and novel result? We argue that the FRIDA framework could detect anomalous experiments that can be reported to a human expert for further investigation. This way, the robot can improve the scale at which experiments are conducted, but valuable, surprising results will not go unnoticed.

8.4 Acknowledgments

This work was only possible through the generosity of the many people who took chances on me along with the incredible collaborators that I have had the pleasure of learning from during my research career.

The following list of acknowledgments does not even begin to show how grateful I am to the people in my life.

To my partner, coauthor, co-conspirator, co-cat-parent, Abena Boadi-Agyemang. Thank you for the years of support. For the laughs after a night of working hard on paper deadlines. For pancakes and weekend skating to bring light into our lives. Thank you to Millies for providing the ice cream that first brought us together at orientation. I love you, Bean.

Committee

I cannot begin to express my gratitude to **my advisor, Professor Jean Oh**. Very few people have the vision that you do, to see the importance and value of the work in this thesis. Thank you for giving me the opportunity to work with you so long ago and sticking with me this whole time. Jean gave incredible guidance on all this work. Without her, the quality would not be close to what it is and likely would not have happened at all. Beyond work, Jean has been an incredible supporter for me personally, and I'll never forget the support she gave when my father passed during my studies. The academic world is much brighter and more fun with you in it. Thank you, Jean.

Thank you to **Professor Jim McCann**, who has been an awesome collaborator for many years. Your approach to robotics is creative and inspiring. Thank you for giving some of the most down-to-earth guidance I have received.

Professor Manuela Veloso, thank you for your guidance through this thesis work as well as your continued support for robot painting. Projects like robotic soccer were an inspiration and paved the way for projects like FRIDA. You taught the first graduate course I ever took. The class kicked my butt, but I have never looked back on researching Machine Learning after that.

Professor Ken Goldberg, as one of the few people to have such success in both engineering and art, I have been inspired by your abilities and achievements. Thank you for always championing FRIDA and finding a place for art & robotics in scientific outlets. Your honest feedback and guidance have been invaluable to me, thank you.

CMU Human-Computer Interaction Institute

Thank you to those in the HCII who gave me my start in research, in particular Cindy Tipper, Jonathan Sewall, and Kenneth Holstein who took chances on an inexperienced student. Thank you to Kenneth Koedinger, Franceska Xhakaj, Hui Cheng, and John Stamper.

Close Collaborators

Zhixuan Liu, who was such a pleasure to work with from her undergraduate summer project on Gander, to StyleCLIPDraw, to SCoFT. Your hard work and authenticity are inspiring.

Vihaan Misra, who pushed so hard on Robot Synesthesia to turn it into the incredible work it is today. A reliable and brilliant friend and collaborator.

Lawrence Chen, who took Spline-FRIDA from a side project to a robust, well-motivated system.

Gaurav Parmar, whose experience and knowledge pushed CoFRIDA to be a strong, clear work.

Jun-Yan Zhu, whose work had insprired me for a long time. It was a pleasure to finally work with you and learn from you.

Thank you to Tanmay Shankar, who helped me in my earliest days joining the roBot Intelligence Group to the end.

Uksang Yoo who participated in incredibly productive yap sessions and taught me all the robotics lore. You and Jeong Hun (JJ) Lee, made those late night paper deadlines fun.

Thank you to Beverley-Clair Okogwu, Wenxuan Peng, Lorie Chen, Yejin Kim, and Jiaying Wei for being incredible collaborators.

Thank you to my lab mates in the roBot Intelligence Group who supported me countless times through the years: Xinjie (Abby) Yao, Jonathan Francis, Eliot Xing, Ingrid Navarro, Sunyu Wang, Ben Stoler, Arthur Bucker, Pablo Ortega Kral, Gavin Zhu, Haokun Zhu, Chengyang Zhao, Hyun Woo Parke, Alonso Cano, Seonmi Park, Al Hassan, Andrew Hundt, Pragna Mannam, Adam Hung, Tai Inui, Lia Coleman, Sam Park, Natasha Mutangana, Chaitanya Chawla, Allen Chang, and Soonmin Hwang.

School Friends Thank you to all the brilliant people I have met through the PhD program who have supported me. I am lucky to call you all friends: Akhil Padmanabha, John Zhang, Bart Duisterhof, Michelle Zhao, Rohan Choudhury, Sarvesh Patil, Sam Speer, Mateo Guaman Castro, Sarah Costrell, Angela Chen, Lisa Egede, Adinawa

Adjagbodjou, Sofia Kwok, Jon Arrizabalaga, Fausto Vega, Aaron Trowbridge, Tejus Gupta, Pranav Khadpe, Yorai Shaoul, Mohamad Qadri, Mrinal Verghese, Thomas Wei.

AI-CARING Thank you to the National Science Foundation for funding this work through the AI-CARING grant. Thank you to Sonia Chernova, Reid Simmons, and other principal investigators for creating an incredible ecosystem for research. Thank you to the many folks I have met and worked with through this grant including Roshni Kaushik, Rayna Hata, Zhi Tan, and Daphne Chen.

Funding Thank you to the organizations who have helped fund this research. The Technology Innovation (Program Meta-human: a virtual cooperation platform for a specialized industrial services) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea). AI-CARING (NSF IIS-2112633). JP Morgan Chase.

Appendices

A Artistic Merit

This thesis primarily presents scientific contributions to the field of robotics; however, the domains (i.e. painting and sculpting) used for developing and evaluating these contributions are inherently artistic. In this section, we discuss the artistic merits and intentions of the pieces created during the process of this thesis.

A.1 Artistic Paintings

The material for the paintings was often chosen to showcase the abilities of the robot. For example, in Figures 4.1 and 4.12, the inputs were selected to show the robot's capabilities. But in examples, such as 4.11, the text "Albert Einstein Dancing" was selected because it is a challenging concept to draw and this would show the need for dynamic re-planning throughout the painting process. Similarly, the collaborative paintings in Figure 6.7 were intended to be challenging for the robot to complete which could show in what was the robot is successful and where it struggles.

The paintings may sometimes involve personal decisions from the authors. For example, "A sad frog ballerina doing an arabesque" was chosen as a text prompt for Figure 6.9 because of my interest in this concept. This was a fitting concept for some feelings I wanted to express, but it was also a challenging prompt that showcased the abilities of the system. It is also nice to pay tribute to artists of the past, such as Tina Turner in Figure 6.10 and Andy Warhol 6.11.

Outside of the academic papers, we used the robots for more artistic purposes. We wanted to combine the exhibition of the robots skills with timely topics to provoke audiences. Following the United States Supreme Court overruling of Roe v Wade, we

Figure A.1: Robot paintings in support of women's rights in the United States of America

made paintings in support of women's rights, displayed in Figure A.1. These were AI generated images which often showed biases against women. In the bottom left example, the robot was prompted to depict supreme court justices, but only male presenting people were generated. In the top middle example of Figure A.1, the AI generated an image of "men oppressing women in America", which created an image of two women reaching out to each other, but their arms lacked hands and they were not able to touch.

While the women's rights paintings were not nearly as powerful as those made by artists and activists who dedicate themselves to the message, they were fulfilling and interesting ways to use research to try to make positive change. These paintings involved AI-generated images. We continued to explore the understanding of the image generators in Figure A.2, where the robot painted images from emotional text prompts. Emotions are purely human, so the AI understanding could only come from the expressions people have created. These paintings served to provoke viewers into an uncanny emotional experience having the emotions of people processed and reflected back at them.

There are many purposes of art in addition to conveying powerful messages or showing technical abilities. Art can be fun and inspiring too. We have conducted dozens

Figure A.2: Paintings of Emotions. In this series, we explored the emotional understanding of text-to-image models in 2021. The robot painted these generated images to provoke an uncanny emotional connection with an AI agent.

of demonstrations of our robot at venues including conferences, grade schools, and community events. Young children are especially inspired when they see a robot that is painting. Students can envision that a career in STEM can touch upon subjects that they love if they were not interested in many of the mainstream robotic domains.

A.2 Artistic Sculptures

In sculpting, many of the examples were chosen because of their ties to previous work. I speculate that the examples, such as alphabetic letters and simple shapes, were chosen by authors of previous work because they are very recognizable. Keeping this tradition, I used the alphabet in Figure 7.1, but added a new twist by using serif font letters. Prior work was simply trying to make a recognizable shape, but with our work we wanted to showcase the details the robot could make in the letter script.

We wanted to make sculptures of detailed source materials, such as making sculptures of people's faces, to send powerful messages. In Figure A.3, we show some of our attempts. The robot was able to sculpt objects that resembled faces or ears, but they were not very recognizable, which limited what we were able to create.

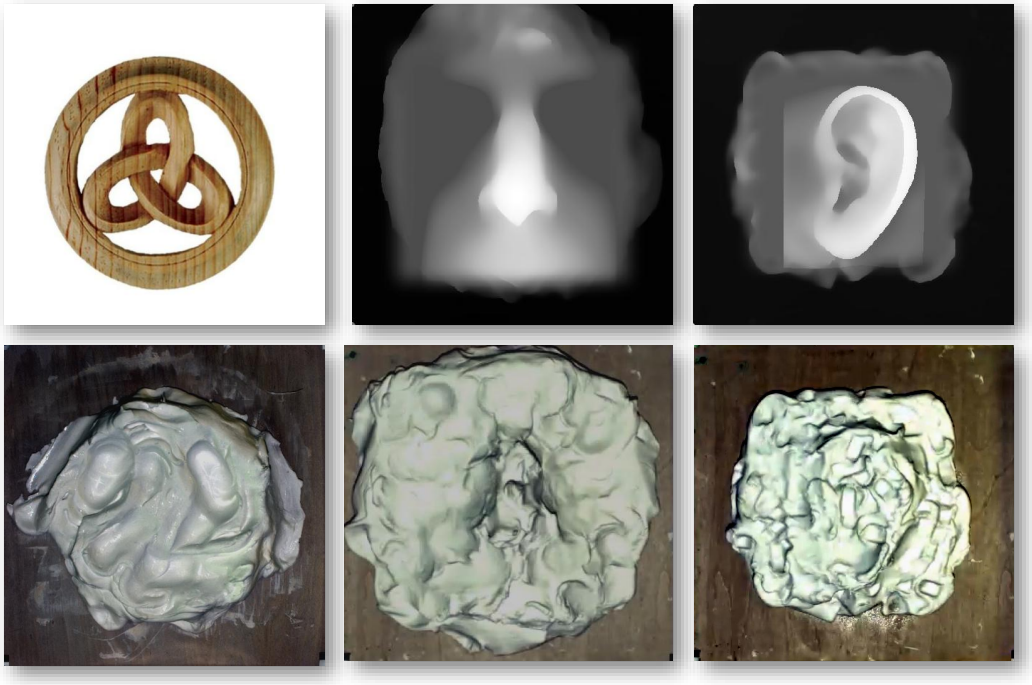

Figure A.3: Sculptures. The top row shows the source material and the bottom row is a photograph of the sculpture the robot made into air-dry foam clay. The top left example was an RGB image that was used as a source where as the right two examples were from 3D model sources.

While faces can be represented well in 2D paintings, we were inspired to make sculptures that had more inherent 3D properties. Inspired by heritage, we made Celtic knots (left example in Figure A.3), which are intricate interwoven patterns that symbolize unity and eternity. In contrast to a 3D printer could easily create a perfect looking Celtic knot, our robot's creation was imperfect and choppy looking. **The continuous lines of Celtic knots represent the endless cycle of life, death, and rebirth, or the eternal nature of things like loyalty, faith, friendship, and love. The robot's imperfect lines symbolize the shortcoming of true friendship, love, or loyalty that robots can provide.**

A.3 Towards Professional, Museum-Worthy Paintings

Art is something that anyone can engage in, since everyone is unique and has something to say creatively. However, art is also an industry and a professional endeavor that people dedicate their entire lives to. It is subjective, but it is safe to say that the

quality of art that makes its way into museums and galleries is much higher than that created by our robot. There is some interesting novelty with our robot painter that separates it from human made work and previous painting projects, but the novelty that human artists can create with their own hands while painting is much better. Additionally, the content of our paintings is generally not very provocative or interesting compared to that of contemporary artists.

Robot paintings have previously made their way into museums and galleries, such as Harold Cohen's AARON at the Whitney and "Power and Water" by Ken Goldberg and Margaret Lazzari at the Fisher Gallery. In these cases, there were powerful concepts being explored and/or great control over the robot such that the human artist was able to convey their ideas. To enable our robot to help in the creation of museum quality artwork, control adjustments would need to be made. People are incredible at creating powerful messages, but in its current state, I believe it would be challenging to use the robot to communicate these messages. With more interaction and input abilities, the robot could be more controllable by a human artist. This may involve the robot working with new materials, having more flexible actions (e.g., blending paint colors), and different types of inputs from the human.

Bibliography

[1] Harry H. Jiang, Lauren Brown, Jessica Cheng, Mehtab Khan, Abhishek Gupta, Deja Workman, Alex Hanna, Johnathan Flowers, and Timnit Gebru. Ai art and its impact on artists. In Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society, AIES '23, page 363–374, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702310. doi: 10.1145/3600211.3604681. URL https://doi.org/10.1145/3600211.3604681.

[2] Cole Bateman. Creating for creatives: A humanistic approach to designing ai tools targeted at professional animators. Bachelor's thesis, Harvard University, 2021.

[3] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In International Conference on Machine Learning, pages 8821–8831. PMLR, 2021.

[4] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[5] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988, 2022.

[6] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 300–309, 2023.

[7] Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. Text-to-3d using gaussian splatting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 21401–21412, 2024.

[8] Aaron Hertzmann. Can computers create art? In Arts, volume 7, page 18. MDPI, 2018.

[9] Aaron Hertzmann. Useful ways to talk about of art (definitions, part 2). Blog, September 2022. URL https://aaronhertzmann.com/2022/09/19/art-definitions-2.html.

[10] Aaron Hertzmann. Confusing definitions of art (definitions, part 1). Blog, September 2022. URL https://aaronhertzmann.com/2022/09/19/art-definitions-1.html.

[11] Uksang Yoo, Adam Hung, Jonathan Francis, Jean Oh, and Jeffrey Ichnowski. Ropotter: Toward robotic pottery and deformable object manipulation with structural priors. In 2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids), pages 843–850. IEEE, 2024.

[12] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. The International Journal of Robotics Research, 43(4):533–549, 2024.

[13] Licia He. Eyeo 2022 - ignite! - licia he. https://vimeo.com/777104438, 2023.

[14] Gerry Chen, Sereym Baek, Juan-Diego Florez, Wanli Qian, Sang-won Leigh, Seth Hutchinson, and Frank Dellaert. Gtgraffiti: Spray painting graffiti art from human painting motions with a cable driven parallel robot. In 2022 International Conference on robotics and automation (ICRA), pages 4065–4072. IEEE, 2022.

[15] Rob Carter and Nick Carter. Dark factory portraits. http://www.robandnick.com/dark-factory-portraits, 2017.

[16] Thomas Lindemeier. e-David: Non-Photorealistic Rendering using a Robot and Visual Feedback. PhD thesis, University of Konstanz, 2018.

[17] Robot art competition. https://robotart.org/.

[18] Ardavan Bidgoli, Manuel Ladron De Guevara, Cinnie Hsiung, Jean Oh, and Eunsu Kang. Artistic style in robotic painting; a machine learning approach to learning brushstroke from human artists. In 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 412–418. IEEE, 2020.

[19] Ken Goldberg. The Robot in the Garden: Telerobotics and Telepistemology in the Age of the Internet. Mit Press, 2001.

[20] Yahav Avigal, William Wong, Mark Presten, Mark Theis, Shrey Aeron, Anna Deza, Satvik Sharma, Rishi Parikh, Sebastian Oehme, Stefano Carpin, et al. Simulating polyculture farming to learn automation policies for plant diversity and precision irrigation. IEEE Transactions on Automation Science and Engineering, 19(3):1352–1364, 2022.

[21] Manuela Veloso, Enrico Pagello, and Hiroaki Kitano. Robocup-99: Robot soccer world cup iii. Springer Science & Business Media, 2000.

[22] Catie Cuan, Tianshuang Qiu, Shreya Ganti, and Ken Goldberg. Breathless: An 8-hour performance contrasting human and robot expressiveness. arXiv preprint arXiv:2411.12361, 2024.

[23] Amy LaViers. Robots and dance: a promising young alchemy. Annual Review of Control, Robotics, and Autonomous Systems, 8, 2024.

[24] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems, 35:25278–25294, 2022.

[25] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13142–13153, 2023.

[26] Allyssia Alleyne. A sign of things to come? ai-produced artwork sells for $433 k, smashing expectations. Cable News Network, 2018.

[27] Artur Karimov, Ekaterina Kopets, Sergey Leonov, Lorenzo Scalera, and Denis Butusov. A robot for artistic painting in authentic colors. Journal of Intelligent & Robotic Systems, 107(3):34, 2023.

[28] Mar Canet Sola and Varvara Guljajeva. Dream painter: Exploring creative possibilities of ai-aided speech-to-image synthesis in the interactive art context. Proceedings of the ACM on Computer Graphics and Interactive Techniques, 5 (4):1–11, 2022.

[29] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 6892–6903. IEEE, 2024.

[30] Andrew Goldberg, Kavish Kondap, Tianshuang Qiu, Zehan Ma, Letian Fu, Justin Kerr, Huang Huang, Kaiyuan Chen, Kuan Fang, and Ken Goldberg. Blox-net: Generative design-for-robot-assembly using vlm supervision, physics simulation, and a robot with reset. In 2025 IEEE International Conference on Robotics and Automation (ICRA), pages 15493–15500. IEEE, 2025.

[31] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. Robotics and autonomous systems, 57(5):469–483, 2009.

[32] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In 2018 IEEE international conference on robotics and automation (ICRA), pages 5628–5635. Ieee, 2018.

[33] Uksang Yoo, Jonathan Francis, Jean Oh, and Jeffrey Ichnowski. Kinesoft: Learning proprioceptive manipulation policies with soft robot hands. arXiv preprint arXiv:2503.01078, 2025.

[34] Alison Bartsch, Arvind Car, Charlotte Avra, and Amir Barati Farimani. Sculptdiff: Learning robotic clay sculpting from humans with goal conditioned diffusion policy. In 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7307–7314. IEEE, 2024.

[35] Alison Bartsch, Arvind Car, and Amir Barati Farimani. Pinchbot: Long-horizon deformable manipulation with guided diffusion policy. arXiv preprint arXiv:2507.17846, 2025.

[36] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. pi0.5: a vision-language-action model with open-world generalization. arXiv preprint arXiv:2504.16054, 2025.

[37] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817, 2022.

[38] Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. arXiv preprint arXiv:2507.05331, 2025.

[39] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In Forty-first international conference on machine learning, 2024.

[40] Eliot Xing, Vernon Luk, and Jean Oh. Stabilizing reinforcement learning in differentiable multiphysics simulation. arXiv preprint arXiv:2412.12089, 2024.

[41] Yutaka Matsuo, Yann LeCun, Maneesh Sahani, Doina Precup, David Silver, Masashi Sugiyama, Eiji Uchibe, and Jun Morimoto. Deep learning, reinforcement learning, and world models. Neural Networks, 152:267–275, 2022.

[42] Alejandro Escontrela, Ademi Adeniji, Wilson Yan, Ajay Jain, Xue Bin Peng, Ken Goldberg, Youngwoon Lee, Danijar Hafner, and Pieter Abbeel. Video prediction models as rewards for reinforcement learning. Advances in Neural Information Processing Systems, 36:68760–68783, 2023.

[43] Sreyas Venkataraman, Yufei Wang, Ziyu Wang, Navin Sriram Ravie, Zackory Erickson, and David Held. Real-world offline reinforcement learning from vision language model feedback. arXiv preprint arXiv:2411.05273, 2024.

[44] Zhewei Huang, Wen Heng, and Shuchang Zhou. Learning to paint with model-based deep reinforcement learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 8709–8718, 2019.

[45] Peter Schaldenbrand and Jean Oh. Content masked loss: Human-like brush stroke planning in a reinforcement learning painting agent, 2021.

[46] Zihan Ding and Hao Dong. Challenges of reinforcement learning. In Deep reinforcement learning: fundamentals, research and applications, pages 249–272. Springer, 2020.

[47] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. Journal of Machine Learning Research, 21(181):1–50, 2020.

[48] Zhewei Huang, Wen Heng, and Shuchang Zhou. Learning to paint with model-based deep reinforcement learning, 2019.

[49] Duy Nguyen-Tuong and Jan Peters. Model learning for robot control: a survey. Cognitive processing, 12(4):319–340, 2011.

[50] Ian Lenz, Ross A Knepper, and Ashutosh Saxena. Deepmpc: Learning deep latent features for model predictive control. In Robotics: Science and Systems, volume 10, page 25. Rome, Italy, 2015.

[51] Nickolas Muray. Frida on bench, 1939. Carbon print, 18 x 14 in.

[52] Guillermo Kahlo. Frida on bench, 1926. Frida Kahlo Museum Trust.

[53] Josh Telles. Portrait of david lynch.

[54] Vincent Lim, Huang Huang, Lawrence Yunliang Chen, Jonathan Wang, Jeffrey Ichnowski, Daniel Seita, Michael Laskey, and Ken Goldberg. Real2sim2real: Self-supervised learning of physical single-step dynamic actions for planar robot casting. In 2022 International Conference on Robotics and Automation (ICRA), pages 8282–8289. IEEE, 2022.

[55] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pages 453–460, 1998.

[56] Kun Zeng, Mingtian Zhao, Caiming Xiong, and Song Chun Zhu. From image parsing to painterly rendering. ACM Trans. Graph., 29(1):2–1, 2009.

[57] Aaron Hertzmann. Paint by relaxation. In Proceedings. Computer Graphics International 2001, pages 47–54. IEEE, 2001.

[58] Kevin Frans, LB Soros, and Olaf Witkowski. Clipdraw: Exploring text-to-drawing synthesis through language-image encoders. arXiv preprint arXiv:2106.14843, 2021.

[59] Peter Schaldenbrand, Zhixuan Liu, and Jean Oh. Styleclipdraw: Coupling content and style in text-to-drawing translation. In Proceedings of the International Joint Conference on Artificial Intelligence, 2022.

[60] Zhengxia Zou, Tianyang Shi, Shuang Qiu, Yi Yuan, and Zhenwei Shi. Stylized neural painting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15689–15698, 2021.

[61] Peter Schaldenbrand and Jean Oh. Content masked loss: Human-like brush stroke planning in a reinforcement learning painting agent. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 505–512, 2021.

[62] Ganghun Lee, Minji Kim, Minsu Lee, and Byoung-Tak Zhang. From scratch to sketch: Deep decoupled hierarchical reinforcement learning for robotic sketching agent. In 2022 International Conference on Robotics and Automation (ICRA), pages 5553–5559. IEEE, 2022.

[63] Aaron Hertzmann. Toward modeling creative processes for algorithmic painting. arXiv preprint arXiv:2205.01605, 2022.

[64] Rundong Wu, Zhili Chen, Zhaowen Wang, Jimei Yang, and Steve Marschner. Brush stroke synthesis with a generative adversarial network driven by physically based simulation. In Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering, pages 1–10, 2018.

[65] Zhili Chen, Byungmoon Kim, Daichi Ito, and Huamin Wang. Wetbrush: Gpu-based 3d painting simulation at the bristle level. ACM Transactions on Graphics (TOG), 34(6):1–11, 2015.

[66] Sen Wang, Jiaqi Chen, Xuanliang Deng, Seth Hutchinson, and Frank Dellaert. Robot calligraphy using pseudospectral optimal control in conjunction with a novel dynamic brush model. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 6696–6703. IEEE, 2020.

[67] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. CoRR, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020.

[68] Federico Galatolo., Mario Cimino., and Gigliola Vaglini. Generating images from caption and vice versa via clip-guided generative latent space search. Proceedings of the International Conference on Image Processing and Vision Engineering, 2021. doi: 10.5220/0010503701660174.

[69] Amy Smith and Simon Colton. Clip-guided gan image generation: An artistic exploration. Evo* 2021, page 17, 2021.

[70] Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. Style transfer by relaxed optimal transport and self-similarity. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10051–10060, 2019.

[71] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2414–2423, 2016.

[72] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[73] Yael Vinker, Ehsan Pajouheshgar, Jessica Y Bo, Roman Christian Bachmann, Amit Haim Bermano, Daniel Cohen-Or, Amir Zamir, and Ariel Shamir. Clipasso: Semantically-aware object sketching. arXiv preprint arXiv:2202.05822, 2022.

[74] Rethink sawyer. https://www.rethinkrobotics.com/sawyer.

[75] Tzu-Mao Li, Michal Lukáč, Michaël Gharbi, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. ACM Transactions on Graphics (TOG), 39(6):1–15, 2020.

[76] Jiaying Wei, Peter Schaldenbrand, Joong Ho Choi, Eunsu Kang, and Jean Oh. Collaborative robotic painting and paint mixing demonstration. In Companion Publication of the 2023 ACM Designing Interactive Systems Conference, pages 292–296, 2023.

[77] Duncan Robertson. The dichotomy of form and content. College English, 28(4): 273–279, 1967.

[78] Chipp Jansen and Elizabeth Sklar. Exploring co-creative drawing workflows. Frontiers in Robotics and AI, 8:577770, 2021.

[79] Nicholas Davis, Chih-PIn Hsiao, Kunwar Yashraj Singh, Lisa Li, and Brian Magerko. Empirically studying participatory sense-making in abstract drawing with a co-creative cognitive agent. In Proceedings of the 21st International Conference on Intelligent User Interfaces, pages 196–207, 2016.

[80] Tomas Lawton, Francisco J Ibarrola, Dan Ventura, and Kazjon Grace. Drawing with reframer: Emergence and control in co-creative ai. In Proceedings of the 28th International Conference on Intelligent User Interfaces, pages 264–277, 2023.

[81] Shayla Lee and Wendy Ju. Adversarial robots as creative collaborators. arXiv preprint arXiv:2402.03691, 2024.

[82] Peter Schaldenbrand, James McCann, and Jean Oh. Frida: A collaborative robot painter with a differentiable, real2sim2real planning environment. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pages 11712–11718. IEEE, 2023. Finalist for Best Paper in Deployed Systems.

[83] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Ruifeng Deng, Xin Li, Errui Ding, and Hao Wang. Paint transformer: Feed forward neural painting with stroke prediction, 2021.

[84] Zhengxia Zou, Tianyang Shi, Shuang Qiu, Yi Yuan, and Zhenwei Shi. Stylized neural painting, 2020.

[85] Xu Ma, Yuqian Zhou, Xingqian Xu, Bin Sun, Valerii Filev, Nikita Orlov, Yun Fu, and Humphrey Shi. Towards layer-wise image vectorization, 2022.

[86] Jingwan Lu, Fisher Yu, Adam Finkelstein, and Stephen DiVerdi. Helpinghand: example-based stroke stylization. ACM Trans. Graph., 31(4), jul 2012. ISSN 0730-0301. doi: 10.1145/2185520.2185542. URL https://doi.org/10.1145/2185520.2185542.

[87] Tom S. F. Haines, Oisin Mac Aodha, and Gabriel J. Brostow. My text in your handwriting. ACM Trans. Graph., 35(3), may 2016. ISSN 0730-0301. doi: 10.1145/2886099. URL https://doi.org/10.1145/2886099.

[88] Tzu-Mao Li, Michal Lukáč, Gharbi Michaël, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. ACM Trans. Graph. (Proc. SIGGRAPH Asia), 39(6):193:1–193:15, 2020.

[89] Ajay Jain, Amber Xie, and Pieter Abbeel. Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models, 2022.

[90] XiMing Xing, Chuang Wang, Haitao Zhou, Jing Zhang, Qian Yu, and Dong Xu. Diffsketcher: Text guided vector sketch synthesis through latent diffusion models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, Advances in Neural Information Processing Systems, volume 36, pages 15869–15889. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/333e67fc4728f147d31608db3ca78e09-Paper-Conference.pdf.

[91] Ximing Xing, Haitao Zhou, Chuang Wang, Jing Zhang, Dong Xu, and Qian Yu. Svgdreamer: Text guided svg generation with diffusion model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 4546–4555, June 2024.

[92] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[93] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. Journal of Machine Learning Research, 9:2579–2605, 2008. URL http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[94] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution, 2018. URL https://arxiv.org/abs/1807.03247.

[95] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL https://arxiv.org/abs/1505.04597.

[96] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. URL https://arxiv.org/abs/2302.05543.

[97] Yuyu Lin, Jiahao Guo, Yang Chen, Cheng Yao, and Fangtian Ying. It is your turn: Collaborative ideation with a co-creative robot through sketch. In Proceedings of the 2020 CHI conference on human factors in computing systems, pages 1–14, 2020.

[98] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18392–18402, 2023.

[99] Francisco Ibarrola, Tomas Lawton, and Kazjon Grace. A collaborative, interactive and context-aware drawing agent for co-creative design. IEEE Transactions on Visualization and Computer Graphics, 2023.

[100] Tomas Lawton, Kazjon Grace, and Francisco J Ibarrola. When is a tool a tool? user perceptions of system agency in human–ai co-creative drawing. In Proceedings of the 2023 ACM Designing Interactive Systems Conference, pages 1978–1996, 2023.

[101] Martin Daniel Cooney and Maria Luiza Recena Menezes. Design for an art therapy robot: An explorative review of the theoretical foundations for engaging in emotional and creative painting with a robot. Multimodal Technologies and Interaction, 2(3):52, 2018.

[102] Martin Cooney and Peter Berck. Designing a robot which paints with a human: visual metaphors to convey contingency and artistry. In ICRA-X Robots Art Program at IEEE International Conference on Robotics and Automation (ICRA), Montreal QC, Canada, page 2, 2019.

[103] Martin Cooney. Robot art, in the eye of the beholder?: Personalized metaphors facilitate communication of emotions and creativity. Frontiers in Robotics and AI, 8:668986, 2021.

[104] Shama Zabeen Shaik, Vidhushini Srinivasan, Yue Peng, Minwoo Lee, and Nicholas Davis. Co-creative robotic arm for differently-abled kids: Speech, sketch inputs and external feedbacks for multiple drawings. In Proceedings of the Future Technologies Conference (FTC) 2020, Volume 3, pages 998–1007. Springer, 2021.

[105] Damith Herath, Jennifer McFarlane, Elizabeth Ann Jochum, Janie Busby Grant, and Patrick Tresset. Arts+ health: New approaches to arts and robots in health care. In Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction, pages 1–7, 2020.

[106] David Ha and Douglas Eck. A neural representation of sketch drawings. arXiv preprint arXiv:1704.03477, 2017.

[107] Peter Schaldenbrand, James McCann, and Jean Oh. Frida: A collaborative robot painter with a differentiable, real2sim2real planning environment. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pages 11712–11718. IEEE, 2023.

[108] Jonas Jongejan, Henry Rowley, Takashi Kawashima, Jongmin Kim, and Nick Fox-Gieg. The quick, draw!-ai experiment. Mount View, CA, accessed Feb, 17 (2018):4, 2016.

[109] Devi Parikh and C Lawrence Zitnick. Exploring crowd co-creation scenarios for sketches. arXiv preprint arXiv:2005.07328, 2020.

[110] et al. Juliet Shen. Co-drawings, 2016. URL https://www.codrawseattle.com/.

[111] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pages 740–755. Springer, 2014.

[112] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. arXiv preprint arXiv:2304.02643, 2023.

[113] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718, 2021.

[114] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In International Conference on Machine Learning, pages 12888–12900. PMLR, 2022.

[115] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. arXiv preprint arXiv:2206.10789, 2(3):5, 2022.

[116] Vihaan Misra, Peter Schaldenbrand, and Jean Oh. Robot synesthesia: A sound and emotion guided ai painter. arXiv preprint arXiv:2302.04850, 2023.

[117] Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. Multimodal datasets: misogyny, pornography, and malignant stereotypes. arXiv preprint arXiv:2110.01963, 2021.

[118] Varvara Guljajeva and Mar Canet Sola. Psychedelic forms-ceramics and physical form in conversation with deep learning. In Proceedings of the Seventeenth International Conference on Tangible, Embedded, and Embodied Interaction, pages 1–5, 2023.

[119] Simon Duenser, Roi Poranne, Bernhard Thomaszewski, and Stelian Coros. Robocut: Hot-wire cutting with robot-controlled flexible rods. ACM Transactions on Graphics (TOG), 39(4):98–1, 2020.

[120] Giulio Brugnaro and Sean Hanna. Adaptive robotic carving: training methods for the integration of material performances in timber manufacturing. In Robotic fabrication in architecture, art and design, pages 336–348. Springer, 2018.

[121] Zhao Ma, Simon Duenser, Christian Schumacher, Romana Rust, Moritz Bächer, Fabio Gramazio, Matthias Kohler, and Stelian Coros. Robotsculptor: Artist-directed robotic sculpting of clay. In Proceedings of the 5th annual ACM symposium on computational fabrication, pages 1–12, 2020.

[122] Zhao Ma, Simon Duenser, Christian Schumacher, Romana Rust, Moritz Bächer, Fabio Gramazio, Matthias Kohler, and Stelian Coros. Stylized robotic clay sculpting. Computers & graphics, 98:150–164, 2021.

[123] Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. In Conference on Robot Learning, pages 642–660. PMLR, 2023.

[124] Alison Bartsch, Charlotte Avra, and Amir Barati Farimani. Sculptbot: Pre-trained models for 3d deformable object manipulation. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 12548–12555. IEEE, 2024.

[125] Justin O'Brien and Alan Johnston. When texture takes precedence over motion in depth perception. Perception, 29(4):437–452, 2000.

[126] James T Todd. The visual perception of 3d shape. Trends in cognitive sciences, 8(3):115–121, 2004.

[127] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B. Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In International Conference on Learning Representations, 2021. URL https://openreview.net/forum?id=xCcdBRQEDW.

[128] Mathew Schwartz and Jason Prasad. Robosculpt: Unique molds for design with minimal waste. In Rob— Arch 2012: Robotic Fabrication in Architecture, Art, and Design, pages 230–237. Springer, 2013.

[129] Asena Kumsal Şen Bayram, Emel Cantürk Akyıldız, et al. Clay 3d printing: Exploring the interrelations of materials and techniques. Journal of Design for Resilience in Architecture and Planning, 5(3):314–326, 2024.

[130] Alison Bartsch and Amir Barati Farimani. Llm-craft: Robotic crafting of elasto-plastic objects with large language models. arXiv preprint arXiv:2406.08648, 2024.

[131] Alison Bartsch and Amir Barati Farimani. Planning and reasoning with 3d deformable objects for hierarchical text-to-3d robotic shaping. IEEE Robotics and Automation Letters, 2025.

[132] Lawrence Yunliang Chen, Baiyu Shi, Daniel Seita, Richard Cheng, Thomas Kollar, David Held, and Ken Goldberg. Autobag: Learning to open plastic bags and insert objects. arXiv preprint arXiv:2210.17217, 2022.

[133] Alberta Longhini, Yufei Wang, Irene Garcia-Camacho, David Blanco-Mulero, Marco Moletta, Michael Welle, Guillem Alenyà, Hang Yin, Zackory Erickson, David Held, et al. Unfolding the literature: A review of robotic cloth manipulation. Annual Review of Control, Robotics, and Autonomous Systems, 8, 2024.

[134] Thomas Weng, Sujay Man Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In Conference on Robot Learning, pages 192–202. PMLR, 2022.

[135] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. arXiv:2406.09414, 2024.

[136] Kenneth Yigael Goldberg. Stochastic plans for robotic manipulation. Carnegie Mellon University, 1990.

[137] Hung-Jui Huang, Mohammad Amin Mirzaee, Michael Kaess, and Wenzhen Yuan. Gelslam: A real-time, high-fidelity, and robust 3d tactile slam system. arXiv preprint arXiv:2508.15990, 2025.

[138] Yiren Song, Shijie Huang, Chen Yao, Xiaojun Ye, Hai Ci, Jiaming Liu, Yuxuan Zhang, and Mike Zheng Shou. Processpainter: Learn painting process from sequence data. arXiv preprint arXiv:2406.06062, 2024.

[139] Alexander Leiser and Tim Schlippe. Ai in art: simulating the human painting process. In International Conference on ArtsIT, Interactivity and Game Creation, pages 295–308. Springer, 2021.

[140] Zhixuan Liu, Peter Schaldenbrand, Beverley-Claire Okogwu, Wenxuan Peng, Youngsik Yun, Andrew Hundt, Jihie Kim, and Jean Oh. Scoft: Self-contrastive fine-tuning for equitable image generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10822–10832, 2024.

[141] Zhixuan Liu, Youeun Shin, Beverley-Claire Okogwu, Youngsik Yun, Peter Schaldenbrand, Jihie Kim, and Jean Oh. Culturally-aware stable diffusion: Supporting representation with culturally-aware text-to-image synthesis. 2023.

[142] Margaret A Boden. Creativity and artificial intelligence. Artificial intelligence, 103(1-2):347–356, 1998.

[143] Jiaoping Chen, Laura Brandimarte, and Anjana Susarla. Does increasing reliance on artificial intelligence boost creativity? assessing ai-augmented creativity with large language models. Assessing AI-Augmented Creativity with Large Language Models (July 17, 2024), 2024.

[144] Matthias Griebel, Christoph Flath, and Sascha Friesike. Augmented creativity: Leveraging artificial intelligence for idea generation in the creative sphere. 2020.

[145] Patrícia Alves-Oliveira, Patrícia Arriaga, Ana Paiva, and Guy Hoffman. Yolo, a robot for creativity: A co-design study with children. In Proceedings of the 2017 Conference on Interaction Design and Children, pages 423–429, 2017.

[146] Daisy Fancourt and Saoirse Finn. What is the evidence on the role of the arts in improving health and well-being? A scoping review. World Health Organization. Regional Office for Europe, 2019.

[147] Haishang Wu, Hamid Mehrabi, Panagiotis Karagiannidis, and Nida Naveed. Additive manufacturing of recycled plastics: Strategies towards a more sustainable future. Journal of Cleaner Production, 335:130236, 2022.

[148] Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of novel materials. Nature, 624(7990):86–91, 2023.

[149] Anthony K Cheetham and Ram Seshadri. Artificial intelligence driving materials discovery? perspective on the article: Scaling deep learning for materials discovery. Chemistry of Materials, 36(8):3490–3495, 2024.

[150] Gary Tom, Stefan P Schmid, Sterling G Baird, Yang Cao, Kourosh Darvish, Han Hao, Stanley Lo, Sergio Pablo-García, Ella M Rajaonson, Marta Skreta, et al. Self-driving laboratories for chemistry and materials science. Chemical Reviews, 124(16):9633–9732, 2024.