

Robot Learning Using Multi-Coordinate Elastic Maps

Brendan Hertel and Reza Azadeh

Abstract— To learn manipulation skills, robots need to understand the features of those skills. An easy way for robots to learn is through Learning from Demonstration (LfD), where the robot learns a skill from an expert demonstrator. While the main features of a skill might be captured in one differential coordinate (i.e., Cartesian), they could have meaning in other coordinates. For example, an important feature of a skill may be its shape or velocity profile, which are difficult to discover in Cartesian differential coordinate. In this work, we present a method which enables robots to learn skills from human demonstrations via encoding these skills into various differential coordinates, then determines the importance of each coordinate to reproduce the skill. We also introduce a modified form of Elastic Maps that includes multiple differential coordinates, combining statistical modeling of skills in these differential coordinate spaces. Elastic Maps, which are flexible and fast to compute, allow for the incorporation of several different types of constraints and the use of any number of demonstrations. Additionally, we propose methods for auto-tuning several parameters associated with the modified Elastic Map formulation. We validate our approach in several simulated experiments and a real-world writing task with a UR5e manipulator arm.

I. INTRODUCTION

Robots must be able to provide general use in a wide variety of applications, meaning that they must acquire new skills effortlessly. Pre-programming all necessary movements is infeasible, therefore robots should have methods to learn new skills on-the-fly. One of the most suitable methods for teaching robots new skills is Learning from Demonstration (LfD) [1], where a human teacher demonstrates a skill which is then encoded by the robot. These skills can be performed using a variety of methods: a human user physically moving the robot as in kinesthetic teaching, an operator teleoperating the robot, or a robot watching a human perform the skill. This offers a low-effort and simply programmable method to capture demonstrations which can then be used by a robot learner.

For robots to best reproduce demonstrated skills, they should understand the *meaning* behind them. For some skills, the information present in task space is important, for instance in a reaching skill, arriving at the endpoint is crucial. However, this may not be the case for all skills. When writing a letter, preserving the shape of the letter is more important than the physical position of the letter itself. Therefore, the robot should encode the desired skill in such a way that captures not just the positions of the skill, but also the importance of other features.

Authors are with the Persistent Autonomy and Robot Learning (PeARL) Lab, University of Massachusetts Lowell, Lowell, MA 01854, USA. Emails: brendan.hertel@student.uml.edu, reza@cs.uml.edu

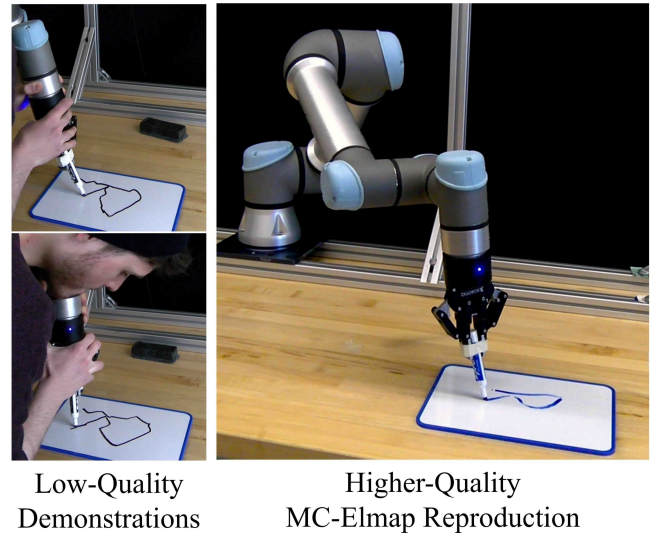


Fig. 1: Multi-Coordinate Elastic Maps (MC-Elmap) performed on a real-world writing skill. Multiple “R” shapes are drawn using a real-world UR5e manipulator arm, but the demonstrations are of low quality as it is difficult to apply enough pressure to write the shape while drawing smoothly. However, MC-Elmap is able to successfully and smoothly reproduce the skill, emphasizing features which were not present in the demonstrations.

Recent advances in the field have introduced a variety of new Learning from Demonstration (LfD) representations. One representation, Multi-Coordinate Cost Balancing (MCCB) [2], encodes demonstrations into multiple differential coordinates, optimizes the importance of these coordinates, and then optimizes a reproduction according to the optimal importance weights. MCCB uses Gaussian Mixture Models and Gaussian Mixture Regression (GMM/GMR) to encode demonstrations. In this work, we use a statistical method which has recently seen many applications in LfD, known as Elastic Maps [3]–[5] to encode demonstrations. Our proposed method, which we denote Multi-Coordinate Elastic Maps (MC-Elmap), provides several advantages over MCCB. First, Elastic Maps have a cost term for smoothness, leading to smoother reproductions in the event of noisy demonstrations, as seen in Fig. 1. Second, Elastic Maps are extremely flexible, and can be used with any number of demonstrations. The contributions of this work are as follows: (i) a novel Elastic Maps formulation which includes the ability to incorporate features from multiple differential coordinates, (ii) methods for auto-tuning hyperparameters, allowing for ease of use and application, and (iii) comparisons with several baselines. We validate the use of MC-Elmap in 28 simulated and one real-world experiment.

II. RELATED WORK

There are many potential approaches to Learning from Demonstration (LfD) [6], all of which can roughly be grouped into four main categories: dynamical systems [7], probabilistic [8], geometric [9], and statistical [10] methods. Each category presents its own unique advantages, and many methods combine these groupings for best results [11]. Dynamical systems attempt to create a stable function over the workspace, resulting in the ability to start from anywhere in said workspace [12]. Probabilistic models find a “likely execution” for a skill and can be highly flexible [13]. Geometric models attempt to capture the geometric features of a skill, which is highly useful for geometrically defined movements, or skills where certain features remain invariant across transformations [14]. Finally, statistical methods find statistical representations of skills, which can often be used as encodings for combining multiple skills or features [15]. In this work, we combine statistical modeling of skills in multiple geometrically-defined differential coordinate spaces, allowing us to capture geometric features across many different demonstrations and skills.

Some LfD methods specifically encode skills in different spaces. One of these methods is Laplacian Trajectory Editing (LTE) [14]. LTE transforms demonstrations into Laplacian coordinate space, adds task constraints, then transforms them back to Cartesian coordinate space. This results in a reproduction which follows the shape of the demonstrations while adhering to any number of initial, final, or via-point constraints. One drawback of this method is that it only uses Laplacian space, and does not consider other coordinate spaces which may capture features of the skill. Additionally, LTE only works with single demonstrations, offering no ability to encode skill variances measured from multiple demonstrations. Building upon LTE, Multi-Coordinate Cost Balancing (MCCB) [2] remedies some of these drawbacks. MCCB uses Gaussian Mixture Models and Gaussian Mixture Regression (GMM/GMR) [10] to model robot skills (although any similar encoding could be used). Additionally, MCCB uses Cartesian, Tangent, and Laplacian differential coordinates, utilizing a meta-optimization to balance the importance of each coordinate according to the skill. Trajectory Learning via Failed and Successful Demonstrations (TLFSD) [15] applies the ideas of MCCB to learn the features specific to success or failure of a skill in multiple differential coordinate spaces. Alternatively, Elastic-Laplacian Trajectory Editing (ELTE) [5] applies the ideas from LTE in an online adaptation scenario, where reproductions must be updated during execution while maintaining the shape of the demonstrations. Our novel approach, MC-Elmap, addresses the main limitations of previous methods by integrating their most essential features. MC-Elmap can incorporate either single or multiple demonstrations, satisfy any number of initial, final, or via-point constraints, automatically determine importance of different coordinate spaces, and be updated for online execution.

Skill learning methods that rely on Reinforcement Learn-

ing (RL) [16] employ an agent which explores different rollouts through the environment, receiving a higher reward for better executions. With RL, the skill must be defined according to specific features, which can include geometric features captured in different coordinate spaces. However, these features may be difficult to define manually, especially with regards to their relative importance in the skill. To this end, Inverse Reinforcement Learning (IRL) [17] offers a solution. In IRL, the features of the skill are predetermined, but their relative importance is unknown. Given some expert examples of the skill, an importance can be found which optimizes the performance of these examples. Similarly to IRL, MC-Elmap uses expert examples to determine the relative importance of features captured in various differential coordinate spaces of a demonstrated skill.

III. METHODOLOGY

A. Background on Elastic Maps

First, we briefly explain Elastic Maps [3], [18]. Elastic Maps model trajectories as a series of nodes connected via springs. Nodes are connected to the demonstration data through springs, as well as adjacent nodes. Adjacent node connections make edges, and pairs of edges create ribs. Minimizing the spring energies associated with the Elastic Map results in an optimized map, which was shown to be well-suited for replicating robot trajectories [3].

Typically, the energies for the Elastic Maps are as follows: (i) the approximation energy $u_{\mathcal{X}}$, which promotes a good fit to the demonstration data (ii) the stretching energy u_E , which promotes equally-spaced nodes, and (iii) the bending energy u_R , which promotes straightness of nodes (the stretching and bending energies together promote smoothness in the result).

B. Multi-Coordinate Elastic Maps

In [5], the approximation energy, $u_{\mathcal{X}}$, was modified such that it approximated the shape of the demonstration rather than converging to the demonstration itself. We take this idea one step further and propose a novel formulation of Elastic Maps inspired by MCCB [2]. In MCCB, the Cartesian (zero-th order), Tangent (first order), and Laplacian (second order) differential coordinate frames of the demonstrations are balanced according to several hyperparameters. We first are given $N \geq 1$ demonstrations as a set $\mathcal{D} = \{\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_N\}$, where a single time-aligned demonstration is defined as $\mathbf{D}_i = [\mathbf{x}_i^1, \mathbf{x}_i^2, \dots, \mathbf{x}_i^T]^\top \in \mathbb{R}^{T \times d}$, with a single data point \mathbf{x}_i^j is a d -dimensional point in space. To convert the demonstrations into different differential coordinates, we use coordinate transformation matrices as

$$\mathbf{D}_{\mathcal{T},i} = \mathbf{T}\mathbf{D}_i, \quad (1)$$

$$\mathbf{D}_{\mathcal{L},i} = \mathbf{L}\mathbf{D}_i, \quad (2)$$

where \mathbf{T} is the Graph Tangent matrix, and \mathbf{L} is the Graph Laplacian matrix, which take the following forms, respectively:

$$\mathbf{T} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix}, \mathbf{L} = \frac{1}{2} \begin{bmatrix} -2 & 2 & 0 & 0 & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & -2 & 1 \\ 0 & \dots & 0 & 0 & 2 & -2 \end{bmatrix},$$

where both matrices are square [2]. Elastic Maps fit all data at once (i.e., batch learning), therefore we concatenate the demonstrations as $\mathbf{g} = [\mathbf{x}_1^1, \mathbf{x}_2^1, \dots, \mathbf{x}_N^T]$ in Cartesian space, and do the same for Tangent and Laplacian spaces, resulting in $\mathbf{g}_{\mathcal{T}}$ and $\mathbf{g}_{\mathcal{L}}$, respectively.

Typically, Elastic Maps use three energy penalty costs in order to find an optimized map: the Cartesian approximation energy, the stretching energy, and the bending energy [3]. These energies are formulated as follows:

$$u_{\mathcal{X}} = w_{\mathcal{X}} \|\mathbf{W}\mathbf{y} - \mathbf{K}\mathbf{g}\|_2^2, \quad (3)$$

$$u_E = \lambda \|\mathbf{E}\mathbf{y}\|_2^2, \quad (4)$$

$$u_R = \mu \|\mathbf{R}\mathbf{y}\|_2^2 \quad (5)$$

where $u_{\mathcal{X}}$ is the Cartesian approximation energy, defined as the difference in Cartesian space between the demonstration data \mathbf{g} and the reproduction $\mathbf{y} \in \mathbb{R}^{M \times d}$. Since $|\mathbf{g}|$ (where $|\cdot|$ represents cardinality) may be greater than $|\mathbf{y}|$, a clustering is needed to determine which data point a node in \mathbf{y} represents. A data point \mathbf{g}_j is clustered to node \mathbf{y}_i if there exists no other node \mathbf{y}_k closer in Euclidean space. This clustering is used to find the clustering matrix \mathbf{K} and weighting matrix \mathbf{W} . The clustering matrix \mathbf{K} is defined such that if data point \mathbf{g}_j is clustered to node \mathbf{y}_i (i.e., $\mathbf{g}_j \in \kappa_i$, where κ_i is the cluster corresponding to \mathbf{y}_i), then $\mathbf{K}_{ij} = 1$, otherwise $\mathbf{K}_{ij} = 0$. \mathbf{W} is a diagonal weighting matrix such that $\mathbf{W}_{ii} = \sum_j \mathbf{K}_{ij}$. The weighting constant $w_{\mathcal{X}}$ defines the importance of the Cartesian approximation energy, and $\|\cdot\|_n$ is the L^n -norm. Additionally, the stretching energy u_E and bending energy u_R find the energies from the edges and ribs, respectively. The edge matrix \mathbf{E} and rib matrix \mathbf{R} are defined as:

$$\mathbf{E} = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & -2 & 1 \end{bmatrix},$$

and the parameters λ and μ are weighting constants for the stretching and bending energies, respectively.

In this novel Elastic Map formulation, we incorporate approximation energy in the Tangent and Laplacian coordinate spaces as well. The energy terms associated with these are as follows:

$$u_{\mathcal{T}} = w_{\mathcal{T}} \|\mathbf{W}\mathbf{y}_{\mathcal{T}} - \mathbf{K}\mathbf{g}_{\mathcal{T}}\|_2^2, \quad (6)$$

$$u_{\mathcal{L}} = w_{\mathcal{L}} \|\mathbf{W}\mathbf{y}_{\mathcal{L}} - \mathbf{K}\mathbf{g}_{\mathcal{L}}\|_2^2, \quad (7)$$

where $u_{\mathcal{T}}$ is the Tangent approximation energy, defined as the difference in Tangent space between the demonstration data and the reproduction, weighted according to the parameter $w_{\mathcal{T}}$, and $u_{\mathcal{L}}$ is the Laplacian approximation energy, defined as the difference in Laplacian space between the demonstration data and the reproduction, weighted according to the parameter $w_{\mathcal{L}}$. The reproduction in Tangent space,

$\mathbf{y}_{\mathcal{T}}$, and Laplacian space, $\mathbf{y}_{\mathcal{L}}$, can be found using the same transformation method as the demonstrations in (1) and (2), respectively.

By minimizing the energy terms in (3)-(7) as

$$\begin{aligned} \underset{\mathbf{y}}{\text{minimize}} \quad & f_0(\mathbf{y}) = \sum_{i \in \{\mathcal{X}, \mathcal{T}, \mathcal{L}, E, R\}} u_i \quad (8) \\ \text{subject to} \quad & f_i(\mathbf{y}) = \|\mathbf{y}_j - \mathbf{z}\|_1 \leq 0 \end{aligned}$$

we find an optimal Multi-Coordinate Elastic Map reproduction. This optimization can include any number of initial, final, or via-point constraints in $f_i(\mathbf{y})$, which are defined such that a point along the reproduction \mathbf{y}_j is constrained to some point in space \mathbf{z} . Additionally, (8) is convex, meaning it can be solved quickly and efficiently [4]. Another benefit of this formulation is that this optimization can also be modified for online trajectory adaptation [5]. Since this optimization incorporates a clustering process, finding an optimal reproduction may change the clustering. Therefore, we can use an Expectation-Maximization (EM) algorithm to iteratively update the clustering (E-step) and solve for an optimal reproduction (M-step) until convergence. Convergence occurs when the clustering does not change between iterations, resulting in a local optimum solution. Pseudocode for this algorithm can be seen in Algorithm 1.

C. Hyperparameter Optimization

Different values of the weighting parameters lead to very different results of (8). Therefore, an optimal set of parameters must be found for each reproduction. To optimize the parameters we perform a meta-optimization on (8). First, the weight values of the approximation terms are estimated as

$$\hat{w}_{\mathcal{X}} = \frac{\alpha_{\mathcal{X}}}{\beta_{\mathcal{X}}}, \quad \hat{w}_{\mathcal{T}} = \frac{\alpha_{\mathcal{T}}}{\beta_{\mathcal{T}}}, \quad \hat{w}_{\mathcal{L}} = \frac{\alpha_{\mathcal{L}}}{\beta_{\mathcal{L}}},$$

where α_i are positive weight values to determine the cost of the corresponding differential coordinate satisfying $\sum_i \alpha_i = 1$; and β_i are scaling factors for the corresponding differential coordinates satisfying $\sum_i \beta_i = 1$. The scaling factors are determined by

$$\beta_i = \frac{\sum_{j=1}^N J_i(\mathbf{D}_j)}{\sum_k \sum_{j=1}^N J_k(\mathbf{D}_j)}, \quad k = \{\mathcal{X}, \mathcal{T}, \mathcal{L}\} \quad (9)$$

where J is the cost associated with the unweighted portion of each approximation energy term, i.e., $J_{\mathcal{X}} = \|\mathbf{W}_i \mathbf{y} - \mathbf{K}_i \mathbf{D}_i\|_2^2$. Note that we use a clustering and weighting matrix for each individual demonstration, denoted \mathbf{K}_i and \mathbf{W}_i respectively, which can be found using the same process as the full clustering and weighting matrices. The cost of each differential coordinate space can be found by minimizing the reproduction error as

$$\{\alpha_{\mathcal{X}}, \alpha_{\mathcal{T}}, \alpha_{\mathcal{L}}\} = \underset{\{\alpha_{\mathcal{X}}, \alpha_{\mathcal{T}}, \alpha_{\mathcal{L}}\}}{\text{minimize}} \sum_{j=1}^N J_{\mathcal{X}}(\mathbf{D}_j) \quad (10)$$

where \mathbf{y} is found by solving (8).

Next, we propose a method for tuning the stretching and bending parameters λ and μ . We base this tuning on the

Algorithm 1: MC-Elmap Expectation-Maximization Algorithm

Input: M , Initial Nodes \mathbf{y} , Data \mathcal{D} , Parameters

 λ_0, μ_0
Output: Optimal Nodes \mathbf{y}^*

```

1 Initialize  $\mathbf{T}, \mathbf{L}, \mathbf{E}, \mathbf{R}$ 
2  $\mathbf{g} \leftarrow \text{CONCATENATE}(\mathcal{D})$ 
3  $\mathbf{g}_{\mathcal{T}} \leftarrow \text{CONCATENATE}(\mathcal{D}_{\mathcal{T}})$ 
4  $\mathbf{g}_{\mathcal{L}} \leftarrow \text{CONCATENATE}(\mathcal{D}_{\mathcal{L}})$ 
5 while not converged do
  // Expectation
6    $\kappa_i = []$  for  $i = 1 \dots M$ 
7   for  $\mathbf{g}_i \in \mathbf{g}$  do
8      $k = \arg \min_{j=1 \dots M} \|\mathbf{g}_i - \mathbf{y}_j\|$ 
9      $\kappa_k \leftarrow i$ 
  // Maximization
10   $\mathbf{K} = []$ 
11   $\mathbf{W} = []$ 
12  for  $i = 1 \dots M$  do
13     $\mathbf{K}_{ij} = 1$  for  $j \in \kappa_i$ 
14     $\mathbf{W}_{ii} = \sum_j \mathbf{K}_{ij}$ 
  // Hyperparameter Tuning
15   $[\beta_{\mathcal{X}}, \beta_{\mathcal{T}}, \beta_{\mathcal{L}}] = \text{CALC-SCALING}(\mathbf{y}, \mathcal{D})$ 
  // eq. (9)
16   $[\alpha_{\mathcal{X}}, \alpha_{\mathcal{T}}, \alpha_{\mathcal{L}}] = \text{minimize} \sum_{j=1}^N J_{\mathcal{X}}(\mathbf{D}_j)$ 
17   $[w_{\mathcal{X}}, w_{\mathcal{T}}, w_{\mathcal{L}}] = [\frac{\alpha_{\mathcal{X}}}{\beta_{\mathcal{X}}}, \frac{\alpha_{\mathcal{T}}}{\beta_{\mathcal{T}}}, \frac{\alpha_{\mathcal{L}}}{\beta_{\mathcal{L}}}]$ 
18   $[\lambda, \mu] = \text{CALC-SMOOTHING}(\mathbf{y}, \mathcal{D}, \lambda_0, \mu_0)$ 
  // eqs. (11), (12)
  // Optimization
19   $\mathbf{y}^* = \text{minimize} \sum_{i \in \{\mathcal{X}, \mathcal{T}, \mathcal{L}, \mathbf{E}, \mathbf{R}\}} u_i$ 

```

assumption that the approximation energies should be about equal to the stretching and bending energies, that is $u_{\mathcal{X}} + u_{\mathcal{T}} + u_{\mathcal{L}} \approx u_{\mathbf{E}} \approx u_{\mathbf{R}}$. Therefore, we use an initial guess for \mathbf{y} and calculate the approximation energies. Then we calculate the bending and stretching energies, unmodified by the constants. To find the parameter λ we apply the following

$$\lambda = \lambda_0 \frac{u_{\mathcal{X}} + u_{\mathcal{T}} + u_{\mathcal{L}}}{\|\mathbf{E}\mathbf{y}\|_2^2} \quad (11)$$

where $\lambda_0 > 1$ is a small constant. The process is the same for finding μ , but replacing \mathbf{E} with \mathbf{R} , as

$$\mu = \mu_0 \frac{u_{\mathcal{X}} + u_{\mathcal{T}} + u_{\mathcal{L}}}{\|\mathbf{R}\mathbf{y}\|_2^2} \quad (12)$$

where $\mu_0 > 1$ is another small constant. The pseudo-algorithm for the Expectation-Maximization algorithm of Multi-Coordinate Elastic Maps (MC-Elmap) along with these parameter tunings (lines 15-18) can be seen in Algorithm 1.¹

IV. EXPERIMENTS

We perform several experiments to evaluate the performance of Multi-Coordinate Elastic Maps (MC-Elmap). We

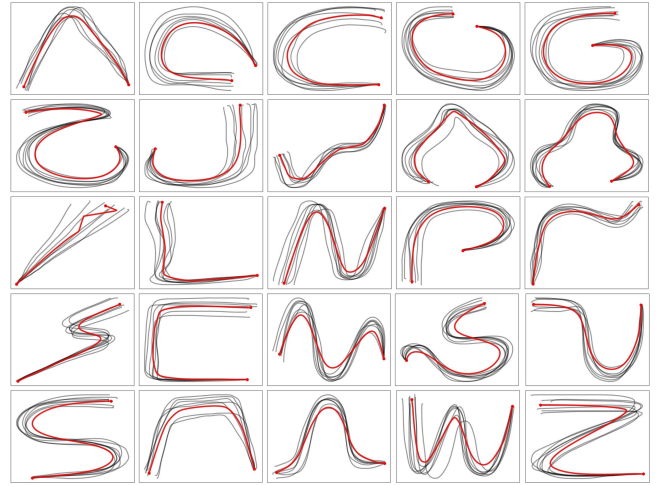


Fig. 2: Results of performing MC-Elmap (red) on the handwriting shapes LASA dataset [12] (demonstrations shown in gray).

first use MC-Elmap on each shape in the LASA dataset [12], a 2D handwriting dataset consisting of 25 shapes (JShape2 has been omitted), shown in Fig. 2. MC-Elmap finds smooth reproductions which maintain the shape of all letters, an important feature in the handwriting task. Additionally, we measure the Fréchet distance, Sum of Squared Errors (SSE), Angular Similarity, and overall jerk of the reproductions (lower is better for all metrics). The Fréchet distance measures spatial but not temporal similarity, SSE measures spatial and temporal similarity, Angular Similarity measures the similarity in shape between two curves, and jerk measures the smoothness of the reproduction. The boxplots of these metrics are shown in Fig. 3, along with comparisons to the following methods:

- Cartesian-only Elmap (Cart) [3] ($w_{\mathcal{X}} = 1, w_{\mathcal{T}} = w_{\mathcal{L}} = 0$),
- Uniform MC-Elmap (Uni) ($w_{\mathcal{X}} = w_{\mathcal{T}} = w_{\mathcal{L}} = 0.33$), and
- MCCB [2].

As can be seen from these comparisons (with mean values across all shapes, normalized by the worst-performing result, listed in Table I), MC-Elmap finds consistently excellent results. As shown in Fig. 3, for each metric, MC-Elmap outperforms other methods. MC-Elmap and MCCB both perform the best on average for the Fréchet distance metric, indicating a strong ability to maintain spatial similarity to demonstrations. This makes sense, as both methods use an optimization which rewards spatial similarity. For SSE, Uniform MC-Elmap performs the best, with automatic weighting MC-Elmap performing the second-best. The reduced performance of auto-weighting could be due to the fact that the auto-weighting favors other factors besides spatial and temporal similarity, such as smoothness. For Angular Similarity, MCCB performs the best, with MC-Elmap showing very similar performance. Finally, MC-Elmap performs the best on jerk, indicating a strong preference for smoothness, which is an important feature for robot trajectories. A comparison between the results of uniform weighting and automatic weighting can be seen in Fig. 4. In this handwriting task,

¹Available at: <https://github.com/brenhertel/MC-Elmap>

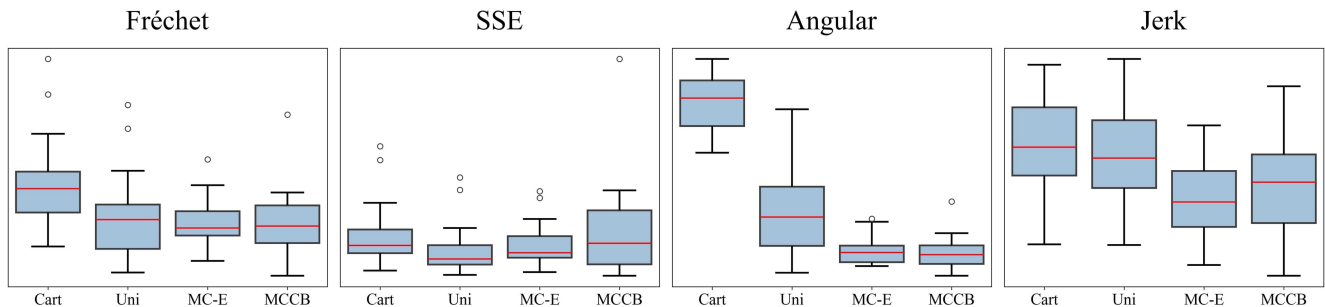


Fig. 3: Boxplots comparing the results of measuring the Fréchet distance, Sum of Squared Errors (SSE), Angular Similarity, and Jerk for different LfD representations on the LASA Dataset (see Fig. 2). MC-Elmap (denoted MC-E in the plots) performs well for all metrics. Boxplot whiskers show 1.5 interquartile range (IQR), with the median shown in red.

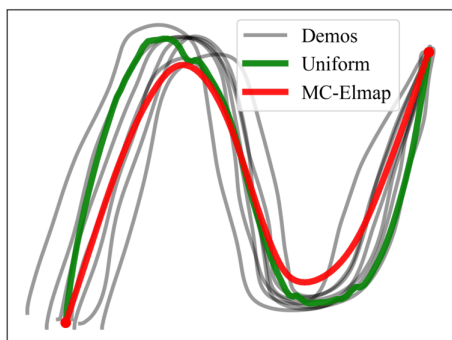


Fig. 4: A comparison between uniform weighting (shown in green) and automatic weighting (shown in red) for MC-Elmap on the “NShape” of the LASA dataset (demonstrations shown in gray). Uniform weighting places a higher importance on Cartesian similarity, leading to some jaggedness in the reproduction in order to increase spatial similarity in the reproduction, whereas the automatic weighting places a higher importance on Laplacian similarity, maintaining the shape of the handwriting skill.

TABLE I: Quantitative metric results from evaluating several methods across all shapes in the LASA dataset (see Fig. 2).

	Cartesian	Uniform	MC-Elmap	MCCB
Fréchet	0.50	0.37	0.35	0.35
SSE	0.20	0.13	0.16	0.22
Angular	0.84	0.36	0.22	0.20
Jerk	0.70	0.73	0.56	0.61

the shape of the demos is most important to the task, and intuitively the Laplacian approximation energy should have more importance. We find that automatic weighting does weight Laplacian approximation more importantly ($\hat{w}_x = 0.01, \hat{w}_T = 0.07, \hat{w}_L = 0.92$), leading to better results than uniform weighting. A high weight for the Cartesian approximation term results in the reproduction introducing jaggedness (as seen around the bends of the “NShape”) which is not seen with automatic weighting.

Additionally, we use MC-Elmap on several 3D robot skills, namely the pressing, pushing, and reaching skills provided by the RAIL dataset [19]. In this dataset, users perform these skills from several starting positions. We highlight the generalizability of this method, showcasing reproductions from several starting positions as seen in Fig. 5. MC-Elmap

is trained on all demonstrations shown, and a reproduction is generated using the initial and final constraints of each individual demonstration. MC-Elmap is able to successfully reproduce all skills from multiple starting points, following trajectories and reproducing the shape of demonstrations while maintaining smoothness. Additionally, MC-Elmap is able to discover important features, such as the first button press in the “pressing” skill, without the need for explicit constraints. In all reproductions, only the start and end point are constrained, although via-points could be used as constraints as well.

Finally, we perform MC-Elmap on a real-world handwriting task using a Universal Robots UR5e manipulator arm (as seen in Fig. 1). This arm was guided using kinesthetic teaching to draw an “R” shape on a whiteboard placed in the robot workspace. Demonstrations were of low quality, resulting in jagged letters, as can be seen from the demonstrations in Fig. 6. This is due to the difficulty of applying enough pressure to write the letter on the whiteboard in combination with the coarse control kinesthetic teaching offers. However, MC-Elmap is able to smooth out the jaggedness, as well as discover the importance of shape for this handwriting task, and successfully reproduce the “R” shape, resulting in a higher-quality reproduction.² This shows MC-Elmap is able to introduce desirable features into reproductions, such as smoothness, which are not necessarily present in demonstrations. Reproductions are computed *a priori* then followed by a low-level controller, with rotations computed using Slerp [20].

V. CONCLUSIONS & FUTURE WORK

In this work we have presented Multi-Coordinate Elastic Maps (MC-Elmap), a method which balances the importance of several differential coordinates, then combines them to find an optimal reproduction. These reproductions can be auto-tuned to incorporate coordinates with more importance without the need for user input. Additionally, the Elastic Map modeling results in a smooth and evenly-spaced reproduction, which is well-suited for robot manipulation. We have presented a novel Multi-Coordinate formulation of Elastic Maps and methods to autotune several parameters associated

²Accompanying video: <https://youtu.be/KU-1dkTa9UE>

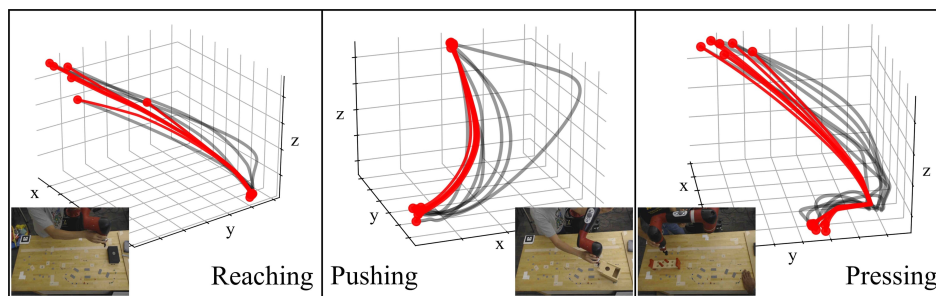


Fig. 5: Results of performing MC-Elmap (red) on the RAIL dataset [19] (demonstrations in gray). A sample real-world demonstration of each skill is shown in the bottom corner for each skill. MC-Elmap is able to successfully reproduce all skills from multiple different starting points, and even is able to reproduce features without constraints, such as the first press in the pressing task.

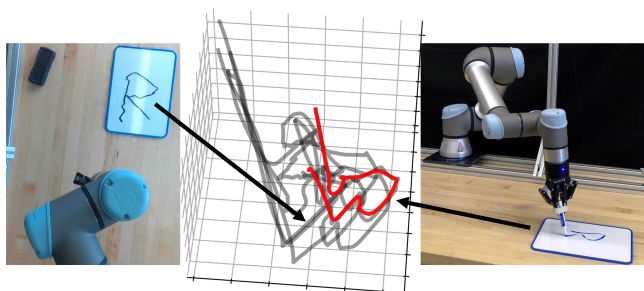


Fig. 6: Results of performing MC-Elmap (red) on a real-world writing task using a UR5e manipulator arm. Demonstrations are shown in gray. Demonstrations complete the task but with undesirable levels of jaggedness. MC-Elmap is able to find a smooth reproduction, resulting in a reproduction which more accurately resembles the intent of the task.

with this formulation. We present this approach with several 2D and 3D datasets, as well as a real-world 3D handwriting task.

There are several avenues for future work. Currently, we use the Cartesian, Tangent, and Laplacian differential coordinates, but this could easily be expanded to include more frames. Alternatively some way of automatically determining the best frames to capture the features of the demonstration could be employed, similar to kernel-based methods. Furthermore, One of the features of Elastic Maps is that it can include variable weighting for each spring used in its energy calculation [18], but here we assume each spring has the same spring constant. Methods could be found to automatically and variably weight the springs, especially in areas of more importance within a skill.

ACKNOWLEDGMENTS

This research is supported in part by the National Science Foundation (FRR-2237463).

REFERENCES

- [1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, 2020.
- [2] H. Ravichandar, S. R. Ahmadzadeh, M. A. Rana, and S. Chernova, "Skill acquisition via automated multi-coordinate cost balancing," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7776–7782.
- [3] B. Hertel, M. Pelland, and S. R. Ahmadzadeh, "Robot learning from demonstration using elastic maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.
- [4] B. Hertel and S. R. Ahmadzadeh, "Confidence-based skill reproduction through perturbation analysis," in *20th International Conference on Ubiquitous Robots (UR)*, 2023, pp. 165–170.
- [5] R. Donald, B. Hertel, S. Misenti, Y. Gu, and R. Azadeh, "An adaptive framework for manipulator skill reproduction in dynamic environments," in *2024 21st International Conference on Ubiquitous Robots (UR)*. IEEE, 2024, pp. 498–503.
- [6] W. Li, Y. Wang, Y. Liang, and D. T. Pham, "Learning from demonstration for autonomous generation of robotic trajectory: Status quo and forward-looking overview," *Advanced Engineering Informatics*, vol. 62, p. 102625, 2024.
- [7] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 763–768.
- [8] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in neural information processing systems*, 2013, pp. 2616–2624.
- [9] S. R. Ahmadzadeh and S. Chernova, "Trajectory-based skill learning using generalized cylinders," *Frontiers in Robotics and AI*, 2018.
- [10] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 37, no. 2, 2007.
- [11] N. Figueroa and A. Billard, "A physically-consistent bayesian non-parametric mixture model for dynamical system learning," in *CoRL*, 2018, pp. 927–946.
- [12] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.
- [13] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, "Kernelized movement primitives," *The International Journal of Robotics Research*, vol. 38, no. 7, pp. 833–852, 2019.
- [14] T. Nierhoff, S. Hirche, and Y. Nakamura, "Spatial adaption of robot trajectories based on laplacian trajectory editing," *Autonomous Robots*, vol. 40, no. 1, pp. 159–173, 2016.
- [15] B. Hertel and S. R. Ahmadzadeh, "Learning from successful and failed demonstrations via optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.
- [16] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 945–990, 2022.
- [17] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the 21st international conference on Machine learning*, 2004, p. 1.
- [18] A. Gorban and A. Zinovyev, "Elastic principal graphs and manifolds and their practical applications," *Computing*, vol. 75, no. 4, 2005.
- [19] M. A. Rana, D. Chen, J. Williams, V. Chu, S. R. Ahmadzadeh, and S. Chernova, "Benchmark for skill learning from demonstration: Impact of user experience, task complexity, and start configuration on performance," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7561–7567.
- [20] K. Shoemake, "Animating rotation with quaternion curves," in *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, 1985, pp. 245–254.