Gaussian Process Dynamical Models for Small-Data Human Motion Synthesis: From Hierarchical Models to Mixture-of-Experts Frameworks

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Jesse Dean St. Amand, M. Sc.
aus Fitchburg, USA

Tübingen
2025

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation: 16.07.2025

Dekan: Prof. Dr. Thilo Stehle
1. Berichterstatter: Prof. Dr. Martin Giese
2. Berichterstatter: Prof. Dr. Daniel Häufle

Disclaimers

Formatting. This thesis uses Felix Dangel's Ph.D. thesis template derived from Federico Marotta's `kaobook` template which was based on Ken Arroyo Ohori's doctoral thesis.

Use of AI. In production of this thesis, I utilized OpenAI's ChatGPT (primarily version o1 [76]) and Anthropic's Claude (primarily version 3.7 Sonnet [3]) for general writing assistance and abstract translation. All AI usage complied with Tübingen University's policy on generative AI. In particular, AI-assisted text was verified for originality through plagiarism checks, and all content derived from AI systems was properly attributed.

Acknowledgments

My years as a PhD student in Tübingen have been transformative, both intellectually and personally. Living in a new country, learning (passable) German, and forming lasting friendships have enriched my life in ways I could not have foreseen.

Foremost, I would like to express my deepest gratitude to my advisor, Prof. Dr. Martin Giese, for his exceptional guidance and unwavering support throughout my PhD journey. He took a chance on a computational neuroscience student venturing into machine learning, granting me the independence and intellectual freedom to develop at my own pace. Transitioning into a new field is challenging, and having a patient, kind mentor to guide me through this process has been an invaluable gift. I am profoundly grateful for his mentorship and support.

I extend my sincere appreciation to my thesis advisory committee members, Prof. Dr. Daniel Häufle and Prof. Dr. Katherine Kuchenbecker, for their insightful feedback and guidance throughout my doctoral research. Their expertise helped ensure my research remained on track, while their career advice proved invaluable for my professional development.

I am deeply thankful to my colleagues in the Computational Sensomotorics Group for their friendship and support. A special acknowledgment goes to Nick Taubert for introducing me to the broader landscape of Gaussian process research and for our many technical discussions. Another special thanks goes to Lucas Martini for editing my German abstract (Zusammenfassung). I would like to also thank additional friends and colleagues from the group: Michael Stettler, Annika Thierfelder, Jens Seemann, Prerana Kumar, Alex Lappe, Xinrui Jiang, Jana Lang, and all other group members who made me feel welcome.

To my friends in Tübingen, particularly those from the DFC community—Lisa Schmors, Alex Meinke, Mila Mollo, Dominic Gonschorek, Wy Ming Lin, Lisa-Marie Elina Erlandsdotter, Abigail Barreiro Pérez, Katja Köhnlein, and many others—thank you for making Tübingen feel like home. Your friendship and support were essential to my well-being during this journey.

I am forever grateful to my family for their unconditional love and support, especially my parents who have encouraged me in all my endeavors. A very special thank you goes to my sister for her unwavering support and acceptance despite the distance that separates us, and my nieces, who bring newfound joy to my life.

The PhD journey is as much an emotional challenge as it is an intellectual one. It's easy to become consumed by the daily grind, pushing oneself to meet deadlines and achieve specific goals. This can lead to unhealthy habits—isolation, perfectionism, poor self-care—and can drain the space meant for other aspects of life. It was the community around me that made the difference, preventing me from becoming isolated and helping me maintain my passion for science and research. For this, I am eternally grateful. Thank you all.

Jesse St. Amand
Tübingen, 2025

Abstract

This thesis details our development of probabilistic architectures based on Gaussian process dynamical models (GPDMs) to address fundamental challenges in human motion modeling. We present a systematic progression from hierarchical GPDMs to the novel Gaussian Process Dynamical Mixture Model (GPDMM), demonstrating how probabilistic approaches can effectively model complex human movements while maintaining interpretability and computational efficiency.

We first establish the effectiveness of hierarchical GPDMs in prosthetic control, showing how these models combine EMG signals with kinematic data to predict hand movements. This work reveals both the potential and limitations of single-class approaches, motivating the development of more sophisticated frameworks for handling multiple movement classes.

Building on these insights, we introduce novel methods for latent space optimization, including the Gaussian process (GP) back-constraint and a GP-based information bottleneck feature selection approach. We then demonstrate that careful initialization with appropriate geometric features can achieve superior performance while simplifying model architecture, challenging conventional approaches that emphasize explicit topological constraints.

The culmination of this work is the GPDMM, which integrates multiple GPDMs in a probabilistic mixture-of-experts framework, enabling both the classification and generation of movement sequences. We demonstrate the GPDMM's viability on these tasks during single-example learning, comparing it with some of the most widely-used deep learning approaches (transformers, VAEs, and LSTMs), which struggled in our data-constrained settings.

Our work establishes that principled probabilistic approaches to motion modeling, like GPDM-based models, can achieve state-of-the-art performance when designed deliberately, maintaining their strengths in interpretability and small-data efficiency. These contributions advance both theoretical understanding and practical implementation in fields ranging from computer animation to prosthetic control.

Zusammenfassung

Diese Dissertation befasst sich mit der Entwicklung probabilistischer Architekturen von Gaußschen Prozessen in dynamischen Modellen (Gaussian Process Dynamical Models, GPDMs), um grundlegende Herausforderungen in der Modellierung menschlicher Bewegungen zu adressieren. Dabei wird eine systematische Weiterentwicklung von hierarchischen GPDMs hin zum neuartigen Gaußschen Prozessen in dynamischen Mixtur-Modell (Gaussian Process Dynamical Mixture Model, GPDMM) herausgearbeitet und gezeigt, wie probabilistische Ansätze komplexe menschliche Bewegungen effektiv abbilden können, ohne dabei an Interpretierbarkeit und Effizienz einzubüßen.

Zunächst zeigen wir die Leistungsfähigkeit hierarchischer GPDMs im Bereich der Prothesenkontrolle, indem wir demonstrieren, wie diese Modelle EMG-Signale mit kinematischen Daten kombinieren, um Handbewegungen vorherzusagen. Diese Arbeit verdeutlicht sowohl das Potenzial als auch die Grenzen von Ansätzen mit nur einer Bewegungs-Klasse, was die Entwicklung anspruchsvollerer Methoden motiviert, um mehrere Bewegungsklassen zu berücksichtigen.

Aufbauend auf diesen Erkenntnissen stellen wir neuartige Verfahren zur Optimierung des latenten Raums des dynamischen Modells vor, darunter die Gaußschen Prozess-(GP-)Back-Constraint-Methodik und einen GP-basierten Informations-Flaschenhals zur Merkmalauswahl. Anschließend zeigen wir, dass eine sorgfältige Initialisierung geeigneter geometrischer Strukturen zu überlegenen Ergebnissen führen kann und zugleich die Modellarchitektur vereinfacht. Damit stellen wir herkömmliche Ansätze in Frage, die explizit topologische Einschränkungen integrieren.

Den Höhepunkt dieser Arbeit bildet das GPDMM, das mehrere GPDMs in einem probabilistischen Mischung-von-Experten System integriert und sowohl zur Klassifizierung als auch zur Generierung von Bewegungssequenzen eingesetzt wird. Wir belegen die Leistungsfähigkeit des GPDMM bei Lernaufgaben von Einzel-Beispielen, indem wir es mit einigen der gängigsten Deep-Learning-Verfahren (Transformern, VAEs und LSTMs) vergleichen, die in unseren datenbegrenzten Szenarien Schwierigkeiten aufweisen.

Unsere Arbeit zeigt, dass fundierte probabilistische Ansätze der Bewegungsmodellierung - wie GPDM-basierte Modelle - bei gezielter Konzeption Höchstleistung erreichen können und zugleich ihre Vorteile in Interpretierbarkeit und Effizienz bei wenig verfügbaren Daten beibehalten. Diese Beiträge erweitern sowohl das theoretische Verständnis als auch die praktische Umsetzung in Bereichen von der Computeranimation bis hin zur Prothesenkontrolle.

Table of Contents

Notation

Mathematical Objects

a	A scalar
$\mathbf{y}$	A column vector representing an observed data point
$\mathbf{Y}$	A matrix of observed data points
$\mathbf{x}$	A column vector representing either latent variables (in latent models) or input variables (e.g. in GPs), depending on context
$\mathbf{X}$	A matrix of either latent variables or input variables, depending on context
$Y_{i,j}$	The (i, j)th entry of matrix $\mathbf{Y}$
$\mathbf{Z}$	Matrix of inducing inputs for sparse approximation
$\mathbf{u}$	Function values at $\mathbf{Z}$
$\mathbf{K}$	Kernel/Gram matrix
$\mathrm{Tr}(\mathbf{K})$	Trace of matrix $\mathbf{K}$
ϵ	Gaussian noise

Gaussian Processes

$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$	A function drawn from a GP prior with zero mean and kernel k
$k(\mathbf{x}, \mathbf{x}')$	Kernel/covariance function
$k_{\mathrm{RBF}}(\mathbf{x}, \mathbf{x}')$	Radial basis function kernel
$k_{\mathrm{lin}}(\mathbf{x}, \mathbf{x}')$	Linear kernel
γ_i	Kernel hyperparameters

Dimensions and Indices

N	Number of data points
D	Dimensionality of the observation space
Q	Dimensionality of the input or latent space
M	Number of inducing points
G	Number of movement classes/experts

Probability and Information Theory

$p(\cdot)$	Probability distribution
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution
$I(\cdot, \cdot)$	Mutual information
$d_F(\cdot, \cdot)$	Fréchet distance

Acronyms & Abbreviations

E.g. or *e.g.*	For example (*exempli gratia*)
Etc. or *etc.*	And so on (*et cetera*)
I.e. or *i.e.*	That is (*id est*)
W.r.t. or *w.r.t.*	With respect to
EMG	Electromyography
GP	Gaussian Process
GPLVM	Gaussian Process Latent Variable Model
GPDM	Gaussian Process Dynamical Model
GPDMM	Gaussian Process Dynamical Mixture Model
BC	Back-Constraint
HMM	Hidden Markov Model
IB	Information Bottleneck
IBLO	Information Bottleneck Latent Optimization
PCA	Principal Component Analysis
RBF	Radial Basis Function
LSTM	Long Short-Term Memory
VAE	Variational Autoencoder

Part I.

Overview

1. Outline

The goal of my doctoral research was to develop new methods for synthesizing human motion, i.e., to create computational models that both categorize and generate movements while respecting the constraints of human physiology. We focused on two critical requirements: (1) that the model be interpretable so that we understand how it is making its predictions and (2) that it performs well when trained on small data sets.

Contemporary machine learning approaches often overlook these factors, instead prioritizing accuracy and generalizability. For example, neural networks—which have been used extensively for high performance in many domains including motion modeling—are typically trained on vast data sets and contain layers of complexity that obscure their decision-making processes.

Yet small-data methods and interpretability remain essential in domains such as prosthetic control. In these applications, data collection faces inherent constraints: limited access to patients and restrictions on the time patients are willing or able to spend performing experimental tasks. Meanwhile, model interpretability is crucial for effective human-machine interaction, ensuring that clinicians understand how to provide optimal treatment and that users can safely and efficiently control prosthetic devices.

To address these challenges, we employ Gaussian process dynamical models (GPDMs). GPDMs provide an intuitive way to represent human movement by mapping high-dimensional body data into a lower-dimensional pose space, while independently modeling the transitions between poses that define dynamics. Moreover, their non-parametric nature and ability to incorporate prior assumptions about human motion dynamics make them especially well-suited for small data sets, helping to mitigate overfitting—a central concern when data is scarce.

2. Introduction

2.1 The Importance of Human Motion Synthesis Models

Human motion synthesis enables models to generate new, physically plausible movements for applications spanning computer animation, robotics, and medical rehabilitation. While human movement is governed by complex biomechanics and contextual factors, effective synthesis models capture these intricacies with sufficient fidelity to produce realistic, adaptable, and purposeful movement sequences [117].

In computer animation, motion synthesis allows creators to generate fluid, natural-looking sequences from limited data. Examples of such data include sparse keyframes (manually defined poses at select important time points, with large temporal gaps between them) or short motion capture recordings (movement data captured from real performers using specialized sensors to track body positions and joint rotations) [11]. By interpolating between existing motions and extrapolating beyond them, these models enable the creation of diverse, lifelike movements without requiring labor-intensive hand animation or continuous motion capture sessions. As audiences and developers increasingly demand authentic and efficient animation pipelines, advanced motion synthesis becomes ever more valuable.

In robotics, the ability to synthesize human-like motions offers two main advantages. First, robots designed to coexist with or assist humans (such as service robots) can leverage movement generation methods that anticipate human kinematics, enhancing safety and communication in shared spaces [70, 124, 125]. Second, these synthesis techniques can inspire novel robotic movement strategies. Human movements often represent biomechanically optimized solutions to tasks, and synthesizing these patterns can help robots achieve more efficient and graceful motions [47, 53, 91].

Furthermore, motion synthesis models drive sophisticated prosthetic and orthotic devices by applying techniques from robotics and neuromusculoskeletal modeling. These applications aim to enhance user quality of life through natural, intuitive control of artificial limbs [34, 35]. Recent advances have produced devices capable of interpreting user intent from various inputs, including residual limb movements and electromyography (EMG) signals that measure skeletal muscle electrical activity [10, 110]. Modern approaches also accommodate patient-specific movement patterns and capabilities, significantly improving functionality and comfort [37, 89].

2.2 Challenges in Human Motion Synthesis

Despite significant advances in computational methods and sensing technologies, human motion synthesis remains an incompletely solved problem due to several fundamental challenges. These challenges span technical, physiological, and computational domains, making the development of comprehensive solutions particularly complex [74].

The continuous nature of human movement creates an effectively infinite number of possible trajectories for connecting an initial configuration to a desired goal. Even when the set of degrees of freedom is fixed, each degree (e.g. joint) can vary continuously, yielding countless paths to a single end position. This multitude of possible paths further complicates the modeling process, as it is difficult to determine which trajectories best capture "natural" movement or most effectively promote functional goals [40].

Motor redundancy presents another challenge. The human body typically has more degrees of freedom than necessary to perform any given task, resulting in multiple possible solutions for achieving the same goal [40, 54, 57]. Although this concept overlaps with the "infinite paths problem" posed in the previous paragraph, they are not equivalent. For example, if our goal is to make a robotic arm with n joints follow a specific end-point trajectory that effectively needs only m (where $m < n$) joints to accomplish, then the arm has $n - m$ "extra" degrees of freedom to coordinate new solutions to the problem [61]. This redundancy makes it difficult to identify optimal or natural movement patterns and complicates the development of control strategies for artificial systems.

The diversity of human movement adds another layer of complexity. Models must account for numerous movement classes and styles, from basic locomotion to complex manipulative tasks. Individual variations in movement execution, influenced by factors such as physical characteristics, skill level, and personal style, further complicate the modeling process [98]. This variability makes it particularly challenging to develop models that generalize well across different individuals and contexts while maintaining naturalness.

Goal execution presents another fundamental challenge in motion modeling, as models must often not only produce realistic movements, but also complete tasks. For example, in prosthetics applications, the model must infer intent from user input (like EMG signals or limb position) in order to generate appropriate responses in real-time and assist in achieving an objective [104, 109]. This coordination of objectives becomes particularly challenging when similar movements have different end goals, requiring models to maintain multiple possible interpretations until sufficient discriminative information becomes available. The challenge scales with the number of possible objectives the model must manage, as each additional goal state increases the complexity of the task.

The requirement for real-time computation in many applications poses particular challenges. Applications such as prosthetic control or interactive animation demand rapid response times, typically requiring computations to be completed within milliseconds [51, 109]. This constraint often necessitates compromises

between model complexity and performance, potentially sacrificing accuracy or generalization capability for speed.

Balancing these competing technical requirements, in itself, is another challenge. A successful motion model must simultaneously address accuracy while maintaining computational efficiency and speed, achieve broad generalization while enabling individual-specific adaptation, and ensure movement naturalness while optimizing task performance. These trade-offs become particularly critical in real-world applications where compromises between different aspects of performance must be carefully calibrated to meet the specific needs of an application [122]. For instance, a prosthetic control system must handle high-dimensional movement data in real-time while accounting for individual user characteristics and maintaining robust performance across various movement conditions [109]. Similarly, animation systems must generate natural-looking movements that respect physical constraints while allowing for stylistic variations and maintaining computational efficiency [22].

The development of effective approaches to human motion synthesis models requires careful consideration of these fundamental challenges and their interactions, particularly in applications where multiple constraints must be satisfied simultaneously. The next section addresses methods used to address these challenges.

2.3 Approaches to Human Motion Synthesis

Human motion synthesis modeling encompasses a range of strategies devised to capture, simulate, and predict the complexity of the human body's movements. Each approach must balance competing goals such as accuracy, physical realism, adaptability, computational efficiency, and interpretability. This section introduces six major classes of methods—procedural, biomechanical, neuromusculoskeletal, synergy-based, machine learning, and hybrid—each offering distinct sets of advantages and limitations.

2.3.1 Procedural

Procedural (rule-based) approaches rely on explicitly defined instructions or constraints to generate or control motion. Such instructions, often stemming from domain expertise, ensure deterministic outcomes by following pre-programmed "if-then" conditions [5]. This approach can be computationally efficient, as each rule is transparent and straightforward to follow, and making it relatively simple to debug or refine specific behaviors [68]. However, because these methods do not typically adapt to changing conditions, they struggle with unanticipated scenarios or intricate tasks [120]. Highly realistic motions may require numerous rules that become unwieldy as complexity grows [68]. Early legged robots using finite state machines for gait, as well as procedural animation systems in video games, illustrate the effectiveness of rule-based strategies while also highlighting their reliance on manual tuning [56].

2.3.2 Biomechanical

Biomechanical approaches use physics-based models grounded in human anatomy and movement mechanics to both analyze and generate movement [55]. These methods simulate the physical properties and constraints of the musculoskeletal system by modeling elements such as joint kinematics, segment inertias, and tissue elasticity. For motion synthesis, biomechanical models use forward dynamics to predict how a body will move given a set of applied forces or torques [97]. Through techniques like optimization-based control, these models can determine the forces needed to achieve desired movements by formulating and solving objective functions that minimize energy expenditure or maximize performance [50, 97]. These models generate physically realistic movements by solving equations of motion with appropriate initial conditions, constraints, and external forces [50, 55]. This approach ensures that generated motions obey fundamental physical laws, avoiding unrealistic movements that might violate joint limits or biomechanical constraints [50, 55]. They are valued for their interpretability—each parameter maps to a tangible aspect of human

physiology [85]—but they can be inflexible, defining movements with a specific set of parameters for a specific task. Moreover, inaccuracies in parameter estimation may accumulate and diminish the reliability of the results [85], a concern that can limit the scalability of purely physics-based models.

2.3.3 Neuromusculoskeletal

Neuromusculoskeletal modeling extends beyond pure biomechanics by explicitly incorporating aspects of neural control. While biomechanical models generate movement through externally specified forces or optimization approaches, neuromusculoskeletal models simulate the biological control signals that drive movement from within the nervous system. These models integrate three key components: neural control signals, musculotendon dynamics, and skeletal biomechanics, creating a comprehensive framework that bridges the gap between neural commands and physical execution [28, 121]. This approach is particularly valuable for investigating neurological disorders, motor learning processes, and rehabilitation strategies, as it can simulate how alterations in neural signaling affect movement patterns [20, 89]. By modeling the closed-loop interaction between sensory feedback and motor output, these models provide unique insights into adaptation mechanisms and motor control strategies that purely mechanical models cannot capture.

2.3.4 Synergy-Based

Synergy-based (or movement primitives) approaches emerge from neurophysiological theories suggesting that the central nervous system employs a small set of fundamental modules or "primitives" to orchestrate complex movements [41]. These techniques typically condense high-dimensional data into a handful of latent factors, which can be recombined in different proportions to generate a wide range of motor behaviors [18, 77, 96]. By highlighting low-dimensional structures, synergy-based models can offer insights into motor coordination patterns and help explain how humans handle the inherent redundancy of the musculoskeletal system [77, 96]. However, their focus on universal building blocks can limit their ability to generalize to new movements, particularly when the number of synergies is limited or these movements are specific to an individual or a specialized task [1, 48]. Additionally, factorization methods that produce these low-dimensional representations, such as principal component analysis or non-negative matrix factorization, can also be sensitive to the amount, quality, and representativeness of the input data.

2.3.5 Machine Learning

Machine learning approaches model motion patterns directly from empirical observations, such as motion capture or wearable sensor data [44, 104]. By processing data sets, these methods adapt to varied contexts [84], individual differences [82], or even entirely new tasks [46, 123]. Neural networks and Gaussian processes, for instance, can uncover features of the movements that underlie how movement is produced, like the constraints imposed by our joints and their simultaneous coordination [112, 119]. This capacity for extracting and applying unintuitive aspects of motion often surpasses that of purely procedural or biomechanical methods, but it comes with substantial data requirements [84]. Collecting large, high-quality motion data sets can be costly [44], learning them can be computationally intensive, and the resulting models may function as "black boxes," offering limited interpretability regarding the physiological mechanisms involved [73, 118]. Despite these challenges, the evolving capabilities of machine learning have made it increasingly central to both academic research and industry applications.

2.3.6 Hybrid

Hybrid approaches combine aspects of procedural, biophysical, or machine learning methods to harness the benefits of each. Biomechanical, neuromusculoskeletal, and synergy-based models can be considered forms of hybrid models, as they often inter-combine methods and incorporate machine learning approaches to learn

biophysical parameters [55, 85, 89]. A hybrid system might, for instance, overlay a machine learning module onto a biomechanical core, ensuring that physical constraints remain intact while still allowing the model to capture the subtleties of real-world variability [30, 52, 90]. Alternatively, it could merge rule-based logic with a learned component that enables more flexible and dynamic application of those rules [2]. Although such combinations can be powerful, they also introduce new layers of complexity. Aligning parameters between disparate modeling paradigms requires careful calibration, and the computational overhead may surpass that of more straightforward single-method approaches.

2.3.7 Selecting an Approach

Each of these approaches—procedural, biomechanical, neuromusculoskeletal, synergy-based, machine learning, and hybrid—addresses different facets of the human motion modeling challenge. Rule-based methods are easier to interpret and implement but less flexible when facing complex or unexpected situations. Biomechanical models boast physical realism yet require extensive parameter estimation and often come with significant computational costs. Neuromusculoskeletal models offer detailed insights into the interplay between neural control and physical execution, though they require sophisticated modeling of both neural and biomechanical components. Synergy-based techniques offer biologically grounded explanations for movement structure, although they may struggle to capture fine-grained details of individual variability. Hybrid models attempt to merge the strengths of multiple frameworks, at the expense of more intricate implementation requirements.

Among these methods, machine learning approaches have seen rapid growth in popularity, thanks to their capacity to learn sophisticated motion patterns and adapt to a wide range of contexts. However, these techniques can be computationally expensive and also introduce domain-specific challenges, including data acquisition and curation demands, and concerns about interpretability. The next section addresses these data challenges, followed by a section on model interpretability.

2.4 The Drawbacks of Data-Centric Approaches

A key limitation of data-centric human motion modeling approaches (e.g., machine learning) lies in the acquisition and curation of the data itself. Modern models often require vast amounts of training data, but in many cases, collecting large-scale human motion data sets is prohibitively expensive or altogether impractical [99]. This section examines the challenges of data collection and curation, highlighting the benefits of developing models well-suited to small data sets.

2.4.1 Challenges in Data Collection and Pre-Processing

Acquiring high-quality human motion data presents substantial challenges. Motion capture systems require meticulous calibration, controlled environments, and extensive pre-processing. Infrared motion capture, while offering excellent temporal and spatial resolution, incurs significant financial costs, restricts movement to camera-visible areas, and demands considerable setup and data cleaning efforts. Electromyography (EMG) recordings face additional complications from electrode placement variations, skin conditions, and other environmental interferences all contributing to signal noise [81]. The pre-processing pipeline—filtering noise, normalizing joint angles, and synchronizing multiple sensor streams—introduces further complexity and time investment. Furthermore, accessing and coordinating with specialized participants, such as specific patient populations or elite performers, can further complicate the process, slowing it down and limiting the amount of data that can be collected [80].

2.4.2 Data Complexity and Size

Human motion data is inherently complex and high-dimensional, often involving dozens of degrees of freedom. High-resolution motion capture systems sample at hundreds of frames per second, producing massive data streams even during short recording sessions [71, 81]. Many applications also require additional measurements, such as force or muscle activation, or the computation and storage of derivatives like velocity and acceleration. Participant-specific factors (e.g., style or skill level) and session-to-session differences (fatigue, accumulated practice, etc.) introduce yet another layer of variability. Collectively, these characteristics pose significant challenges for efficient data storage and processing, especially in real-time applications.

2.4.3 Small-Data Advantages

While small data sets may seem limiting, they offer several distinct advantages beyond reducing collection and processing costs. The manageable size enables developers to carefully curate and fine-tune the data for specific applications, leading to higher quality training sets with improved signal-to-noise ratios that can lead to better model performance [8]. This intimate familiarity with the data also allows for a more informed design of model constraints and priors. Furthermore, smaller data sets also facilitate clearer diagnosis of issues with model behavior and a more straightforward interpretation of trends that may stem from fine-grained aspects of the data distribution.

2.5 The Critical Role of Interpretability in Motion Synthesis

Interpretability is a pivotal concern when developing human motion synthesis models, particularly in fields like medicine and robotics where the close interaction between humans and machines is critical. While black-box deep learning methods often deliver impressive predictive power, their opaque nature can hinder clinical decision-making, user trust, and refinement of movement algorithms [75]. In many high-stakes domains, experts must understand the reasoning behind a model's outputs to ensure safety, regulatory compliance, and alignment with real-world physiology [88].

2.5.1 Rehabilitation

Rehabilitation technologies increasingly incorporate AI-driven motion synthesis across a spectrum from restorative approaches (aiming to return patients to pre-injury function) to compensatory strategies (developing alternative movement patterns). Interpretability serves distinct purposes for clinicians and end-users: healthcare professionals require transparency into therapeutic reasoning, confidence levels, and biomechanical assumptions to validate clinical alignment and customize protocols [17], while patients benefit from accessible explanations that build motivation and autonomy without overwhelming technical detail [14].

Two key examples illustrate these differing needs. Myoelectric prostheses represent compensatory rehabilitation, where clinicians need detailed signal processing metrics to optimize device function. Meanwhile, users of these prostheses require intuitive feedback about control capabilities to effectively utilize the technology [14, 17]. Robotic exoskeletons, as restorative interventions, present a different case. Here, therapists must visualize the relationship between detected muscle activity and machine assistance to accurately assess recovery progress. Simultaneously, patients using exoskeletons benefit from simplified feedback about their contributions to movement, serving as motivational indicators during rehabilitation [16].

2.5.2 Human-Robot Collaboration

Interpretability also proves critical in industrial human-robot interaction, where collaborative robots share workspaces with human operators. A motion synthesis model might govern the robot's path planning or anticipate human actions to avoid collisions [72]. However, if the model's decision-making process is opaque, a sudden, unexplained move could alarm operators or cause accidents. In contrast, an interpretable framework–one that highlights, for instance, its estimates of human limb trajectories–can enable workers and supervisors to anticipate and adapt to the robot's actions. This mutual understanding not only improves safety but also streamlines workflow, as humans can trust that the robot will predictably respond to dynamic shop-floor environments [31, 60].

2.5.3 Regulatory and Safety Requirements

Regulatory frameworks increasingly mandate interpretability as a core safety requirement across both rehabilitation technologies and industrial human-robot collaboration. Industrial safety standards like ISO/TS 15066 requires predictable robot behaviors in collaborative settings [87] and ISO/IEC standards on artificial intelligence (e.g., 42001, 23894) advocate for the use of transparent and explainable systems [92]. Beyond compliance, interpretability can build essential trust among diverse stakeholders—clinicians, patients, and insurers in rehabilitation contexts; workers, supervisors, and safety officers in industrial settings. As these technologies advance toward greater autonomy in increasingly intimate human contexts, incorporating interpretability into design processes from the earliest stages becomes not merely advantageous, but essential for market access and implementation.

2.6 Model Selection: The Case for Gaussian Process Dynamical Models

In data-limited or safety-critical scenarios, probabilistic approaches to human motion modeling offer distinct advantages. These methods provide principled uncertainty management while allowing for the incorporation of domain knowledge like the expected behavior of a dynamical system or biomechanical constraints. When large-scale data collection is impractical or prohibitive (as discussed in section 2.4), probabilistic models can operate effectively with smaller, carefully curated data sets by embedding this domain knowledge—such as dynamics, joint limits, muscle synergies, or energetic costs—directly into the modeling process.

From a Bayesian perspective, these constraints are embedded into the model through probabilistic priors that regularize the parameter space, guiding the model toward expected solutions that the data alone may not account for, all while maintaining a principled structure that can be readily interpreted. This transparent design enables practitioners to trace how different variables or assumptions affect movement predictions—a crucial requirement in fields where interpretability is a core safety concern (see section 2.5). Additionally, probabilistic methods inherently quantify prediction uncertainty, identifying situations where they may be operating outside their reliable range—also especially valuable in safety-sensitive applications.

2.6.1 Gaussian Processes for Human Motion

Among probabilistic approaches, Gaussian processes (GPs) offer a particularly flexible framework for capturing movement patterns. Rather than committing to specific functional forms, GPs maintain distributions over possible functions, naturally regularizing solutions while limiting unwarranted assumptions. This flexibility allows both point estimates via the mean function and plausible motion realizations through function sampling.

Various GP-based approaches have been developed for human motion synthesis. For example:

- **GP regression** maps high-dimensional inputs (e.g., joint angles) to outputs (e.g., subsequent poses), using kernels that encode assumptions about movement smoothness [9].

- **Gaussian Process Latent Variable Models (GPLVMs)** learn low-dimensional embeddings of motion data, compressing many degrees of freedom into interpretable manifolds, revealing underlying structure and simplifying tasks like visualization or synthesis [19, 42, 62].
- **Gaussian Process State-Space Models (GPSSMs)** incorporate temporal dependencies by treating both state-transition and observation functions as GPs. These models capture nonlinear dynamics while remaining interpretable through modular kernel design, making them suitable for adaptive systems requiring real-time updates [33, 59].
- **Gaussian Process Dynamical Models (GPDMs)** are the primary model considered in this thesis, outlined in more detail below and in section 3.3. They extend the GPLVM, integrating dimensionality reduction with temporal modeling, offering a particularly well-suited framework for human motion synthesis [111, 112].

2.6.2 Gaussian Process Dynamical Models

Gaussian process dynamical models (GPDMs) combine the strengths of dimensionality reduction with temporal modeling. GPDMs extend GPLVMs by incorporating a temporal prior over the latent space, modeling state evolution as an autoregressive process [111, 112]:

$$\mathbf{x}_t = f_x(\mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-z}) + \boldsymbol{\epsilon}, \tag{2.1}$$

where $z \in \{1, 2\}$ determines the dynamics order, $\boldsymbol{\epsilon}$ represents Gaussian noise, and $f_x(\mathbf{x}) \sim \mathcal{GP}(0, k_x(\mathbf{x}, \mathbf{x}'))$.

Conceptually, GPDMs can be viewed as continuous-valued, non-linear extensions of hidden Markov models, where state transitions are governed by Gaussian processes instead of discrete transition matrices [111, 112]. This combination captures both manifold structure (through the GPLVM mapping from latent states to observations, the emission GP) and temporal evolution (through the dynamical GP). Moreover, for applications in human motion, GPDMs can be viewed as a method for dividing the problem intuitively into two separate parts: first, capturing representations of the pose of the body at each individual frame in a sequence via the emission GP, and second, modeling the transition between poses via the dynamical GP [111, 112].

Unlike standard GPLVMs, the temporal prior $p(\mathbf{X})$ encourages smooth trajectories that reflect the true dynamical structure of the data. This additional constraint leads to more coherent embeddings for time-series data, preventing discontinuities in the learned latent space [111, 112]. When compared to GPSSMs, GPDMs place greater emphasis on learned latent representations, which makes them particularly effective for high-dimensional motion data where meaningful low-dimensional structure exists.

GPDMs address many key challenges in human motion synthesis:

- Reduction of high-dimensional joint space data to lower-dimensional manifolds that capture essential spatial patterns
- Modeling of temporal dependencies in a Markovian fashion, allowing for flexible transitions characteristic of adaptive human motion
- Quantification of uncertainty in both the dynamic process and observation mapping
- Provision of interpretable representations through the latent space that can be used to address irregularities and guide further refinement of the model

These properties make GPDMs particularly well-suited for applications ranging from character animation to human-robot interaction and rehabilitation robotics—which as stated previously, are domains often constrained by data availability and a need for interpretable models.

A more detailed mathematical treatment of GPDMs is provided in the Background & Methods chapter, section 3.3.

2.7 Research Objectives

Given the challenges and requirements outlined above, this thesis addresses the development of a comprehensive motion modeling framework that simultaneously satisfies multiple critical constraints. Specifically, we aim to design a model capable of both classifying movement intentions and generating appropriate continuing motion while operating under practical limitations common to real-world applications.

The proposed model must accomplish five key objectives. First, it must accurately classify movements from partial observations, enabling early recognition of user intentions. Second, it needs to perform forward prediction of movement trajectories that maintain both accuracy and naturalistic qualities. These two capabilities must be integrated seamlessly, as classification results inform the generation of appropriate continuing motion. Third, the model must perform effectively with limited training data, preferentially learning from single examples of each movement type to accommodate scenarios where extensive data collection is impractical or impossible. Fourth, the model architecture must maintain interpretability, providing insights into its decision-making process and quantifying prediction uncertainties. Finally, the model must achieve all these objectives while maintaining real-time-capable computational performance.

To address these requirements, we design and apply architectural variations of the GPDM over three projects presented in this thesis. The first project introduces a hierarchical GPDM that predicts the target of reach-to-grasp movements and completes the action by generating a sequence of joint angles. The second and third projects introduce the Gaussian process dynamical mixture model (GPDMM), a probabilistic mixture-of-experts model that combines multiple GPDMs into the same latent space. While the second project evaluates an application of the early (unrefined) GPDMM to the information bottleneck principle, the third project formally introduces the model and showcases its utility and some of its variations.

Part II.

Background & Methods

3. Theory

3.1 Gaussian Processes (GPs)

A Gaussian process (GP) can be defined as a stochastic process where any finite collection of random variables exhibits multivariate Gaussian behavior [114]. This mathematical framework enables the representation of probability distributions over functions, providing robust statistical approaches for regression and classification tasks.

3.1.1 Formulation

Suppose we have a training data set,

$$\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N},$$

where each input $\mathbf{x}_i \in \mathbb{R}^Q$ and each observed output $\mathbf{y}_i \in \mathbb{R}^D$. We assume additive Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ on the outputs where σ^2 is the noise variance and $\mathbf{I}$ is the identity matrix. Formally, a GP is specified by a mean function $m(\mathbf{x})$ and the covariance represented by a kernel function $k(\mathbf{x}, \mathbf{x}')$ [114]:

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}),\, k(\mathbf{x}, \mathbf{x}')\big). \tag{3.1}$$

For any finite collection of input points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, the corresponding function values follow a multivariate normal distribution [114]:

$$\big[f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)\big]^T \sim \mathcal{N}\big(\boldsymbol{\mu},\, \mathbf{K}\big), \tag{3.2}$$

where $\boldsymbol{\mu}_i = m(\mathbf{x}_i)$ and $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. In many applications, one adopts a zero mean function ($m(\mathbf{x}) = 0$) without loss of generality, since any known trend can be absorbed into the kernel.

3.1.2 Joint Distribution and Likelihood

Let $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)]^T$ denote the vector of function values at the N training points. Under the GP prior [114],

$$p(\mathbf{f} \mid \mathbf{X},\, \boldsymbol{\theta}) = \mathcal{N}\big(\mathbf{0},\, \mathbf{K}\big), \tag{3.3}$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T$ and $\mathbf{K}$ is the $N \times N$ kernel matrix computed by $k(\mathbf{x}, \mathbf{x}')$. In a regression setting with additive Gaussian noise, each observed $\mathbf{y}_i$ relates to $f(\mathbf{x}_i)$ as [114],

$$p(\mathbf{y}_i \mid f(\mathbf{x}_i)) = \mathcal{N}\big(\mathbf{y}_i \mid f(\mathbf{x}_i),\, \sigma^2\mathbf{I}\big). \tag{3.4}$$

Hence, the joint distribution over $\mathbf{f}$ and the observed outputs $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^T$ factorizes as [114],

$$p(\mathbf{f}, \mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) = p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\theta}) \prod_{i=1}^{N} p\big(\mathbf{y}_i \mid f(\mathbf{x}_i)\big). \tag{3.5}$$

3.1.3 Predictive Distribution

For a new set of inputs $\mathbf{X}_*$, we denote the corresponding function values as $\mathbf{f}_*$. By conditioning on the observed data $\mathcal{S}$ and applying GP properties, one obtains the predictive distribution [114]:

$$p(\mathbf{f}_* \mid \mathbf{X}_*, \mathcal{S}) = \mathcal{N}\big(\boldsymbol{\mu}_*,\, \boldsymbol{\Sigma}_*\big), \tag{3.6}$$

with

$$\boldsymbol{\mu}_* = \mathbf{K}_*^T \left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{Y}, \tag{3.7}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{K}_*. \tag{3.8}$$

Here, $\mathbf{Y} \in \mathbb{R}^{N \times D}$ is the matrix of all observed outputs, $\mathbf{K}$ is the $N \times N$ kernel matrix on the training inputs, $\mathbf{K}_*$ is the $N \times n$ cross-covariance between the training inputs and the n new inputs $\mathbf{X}_*$, and $\mathbf{K}_{**}$ is the $n \times n$ kernel matrix between the new inputs. The term $\sigma^2\mathbf{I}$ accounts for the additive noise variance in each observation.

3.1.4 Kernel Choice

The choice of kernel function is crucial for capturing the desired properties of the underlying function space. In this work, we often use a combination of a radial basis function (RBF) kernel and a linear kernel [114]:

$$k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = \gamma_1 \exp\left(-\frac{\gamma_2}{2} \|\mathbf{x} - \mathbf{x}'\|^2\right), \tag{3.9}$$

$$k_{\text{lin}}(\mathbf{x}, \mathbf{x}') = \gamma_3 \, \mathbf{x}^T \mathbf{x}', \tag{3.10}$$

where γ_i are hyperparameters learned from data. The RBF kernel captures local smoothness, while the linear kernel models global linear trends.

3.1.5 From Observed Inputs to Latent Variables

Thus far, we have assumed that the inputs $\{\mathbf{x}_i\}$ are fully observed. However, in many scenarios, especially when $\mathbf{y}_i \in \mathbb{R}^D$ is high-dimensional, it may be more natural to view these observations as originating from an unknown latent space. The Gaussian process latent variable model (GPLVM) addresses this by treating the inputs $\mathbf{x}_i$ themselves as latent and learning a manifold that explains the data. We will introduce the GPLVM framework in the next section, showing how it extends GPs to unsupervised learning tasks.

3.2 GPLVMs

The Gaussian process latent variable model (GPLVM) extends the Gaussian process (GP) framework to unsupervised learning by treating the inputs as latent variables to be learned along with the GP parameters [62]. Conceptually, the GPLVM can be viewed as a non-linear generalization of probabilistic principal component analysis (PPCA). Indeed, when using a linear kernel, the GPLVM recovers PPCA, but its non-linear kernel formulation enables it to represent more complex manifold structures [62].

Let us consider a data set,

$$\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^T \in \mathbb{R}^{N \times D} \tag{3.11}$$

and let,

$$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N \times Q} \tag{3.12}$$

denote the corresponding *latent* inputs, with $Q \ll D$. Each observed data point $\mathbf{y}_n$ is generated by a latent point $\mathbf{x}_n$ through the mapping [62],

$$\mathbf{y}_d = f_d(\mathbf{x}) + \boldsymbol{\epsilon}_d, \tag{3.13}$$

where $\mathbf{y}_d$ denotes the d-th column of $\mathbf{Y}$, $\boldsymbol{\epsilon}_d$ is Gaussian noise, and $f_d \colon \mathbb{R}^Q \to \mathbb{R}$ is drawn from a GP prior [62]:

$$f_d(\mathbf{x}) \sim \mathcal{GP}\big(0,\, k_f(\mathbf{x}, \mathbf{x}')\big). \tag{3.14}$$

In this thesis, the kernel $k_f(\mathbf{x}, \mathbf{x}')$ is typically a combination of an RBF term (equation 3.9) and a linear term (equation 3.10), plus a bias term γ_i to account for global offsets in the data.

3.2.1 Model Inference

Unlike standard supervised GPs, where the inputs $\mathbf{X}$ are assumed *observed*, the GPLVM treats $\mathbf{X}$ as unknown latent variables. Since $\mathbf{f}$ follows a GP prior and the noise is assumed Gaussian, the *conditional* marginal likelihood of $\mathbf{Y}$ given $\mathbf{X}$ and kernel hyperparameters $\boldsymbol{\theta}$ is tractable. Specifically, for each dimension d [62],

$$p(\mathbf{y}_d \mid \mathbf{X}, \boldsymbol{\theta}) \;=\; \int p(\mathbf{y}_d \mid \mathbf{f}, \boldsymbol{\theta})\, p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\theta})\, d\mathbf{f} \;=\; \mathcal{N}\big(\mathbf{y}_d \mid \mathbf{0},\, \mathbf{K} + \sigma^2 \mathbf{I}\big), \tag{3.15}$$

where $\mathbf{K}$ is the $N \times N$ covariance matrix computed via the kernel $k_f(\mathbf{x}, \mathbf{x}')$ on the latent points $\mathbf{X}$, and σ^2 is the noise variance. Collecting all D output dimensions, the (conditional) log-likelihood of $\mathbf{Y}$ given $\mathbf{X}$ is [62]

$$L = -\frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T\right) - \frac{ND}{2}\ln(2\pi). \tag{3.16}$$

In the GPLVM, the inputs $\mathbf{X}$ are not observed. Rather than attempting to integrate them out to find $\boldsymbol{\theta}$ (which is intractable for general non-linear kernels), we treat them as learnable parameters jointly with the parameters $\boldsymbol{\theta}$. One common strategy is to place a weak prior $p(\mathbf{X})$ on the latent points and perform a *maximum a posteriori* (MAP) optimization [62]:

$$\{\mathbf{X}^*, \boldsymbol{\theta}^*\} = \arg\max_{\mathbf{X},\boldsymbol{\theta}} \Big[\log p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\theta}) + \log p(\mathbf{X})\Big]. \tag{3.17}$$

This formulation allows us to discover a low-dimensional embedding $\mathbf{X}$ while simultaneously adjusting the kernel parameters $\boldsymbol{\theta}$ to fit the data.

Although the prior $p(\mathbf{X})$ can help constrain the embedding to a meaningful region, if it is sufficiently weak, the MAP solution will closely resemble the maximum-likelihood estimate (MLE). This is because, in the limit of uninformative priors (large prior variance), MAP converges to MLE. Note also that equation 3.17 excludes the Bayesian normalizing term $p(\mathbf{Y})$, since it does not depend on $\mathbf{X}$ or $\boldsymbol{\theta}$.

3.2.2 Computational Complexity and Gradients

Maximizing the log-likelihood in Equation equation 3.16 is computationally expensive for large N. Two bottlenecks in each iteration are:

1. Inverting the kernel matrix $\mathbf{K}$, an $\mathcal{O}(N^3)$ operation.
2. Computing the trace term $\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T\right)$, which typically requires $\mathcal{O}(N^2 D)$.

Since these operations appear in both the evaluation of the objective and its gradients with respect to $\mathbf{X}$ and $\boldsymbol{\theta}$, the total cost for T optimization iterations can be on the order of $\mathcal{O}(T\,N^3)$. This quickly becomes infeasible as N grows.

Derivatives with respect to the latent points $\mathbf{X}$ can be computed via matrix calculus; for instance, one obtains [62, 69]:

$$\frac{\partial L}{\partial \mathbf{X}} = -D\,\mathbf{K}^{-1}\mathbf{X} + \mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T\mathbf{K}^{-1}\mathbf{X}, \tag{3.18}$$

although the full expression can involve additional terms if the kernel has explicit dependence on $\mathbf{X}$. Efficiently computing these gradients and dealing with the $\mathcal{O}(N^3)$ scaling remains an active area of research, motivating sparse approximations and other scalability techniques [62, 64, 65, 93, 106]. See the Background & Methods chapter, section 3.5 for more details.

3.3 Gaussian Process Dynamical Models (GPDMs)

A Gaussian process dynamical model (GPDM) extends the GPLVM framework by incorporating a temporal prior over the latent space [111, 112]. Specifically, rather than assuming that latent points $\{\mathbf{x}_t\}$ are *independently* drawn from a prior, we impose a *dynamical* structure to model state evolution over time. This is accomplished by placing a GP prior on the latent variables that defines an autoregressive transition function [111, 112],

$$\mathbf{x}_t = f_x(\mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-z}) + \boldsymbol{\epsilon}, \tag{3.19}$$

where $z \in \{1, 2\}$ determines the order of the dynamics, $\boldsymbol{\epsilon}$ represents Gaussian noise, and,

$$f_x(\mathbf{x}) \sim \mathcal{GP}\big(0,\, k_x(\mathbf{x}, \mathbf{x}')\big). \tag{3.20}$$

When $z = 2$ (i.e., second-order dynamics), the kernel function must capture dependencies on the two previous latent states. Concretely [111, 112]:

$$k_x\big([\mathbf{x}_{t-1}, \mathbf{x}_{t-2}],\ [\mathbf{x}_{\tau-1}, \mathbf{x}_{\tau-2}]\big) = \gamma_5 \exp\!\Big(-\tfrac{\gamma_6}{2}\|\mathbf{x}_{t-1} - \mathbf{x}_{\tau-1}\|^2 - \tfrac{\gamma_7}{2}\|\mathbf{x}_{t-2} - \mathbf{x}_{\tau-2}\|^2\Big) + \gamma_8\, \mathbf{x}_{t-1}^T \mathbf{x}_{\tau-1} + \gamma_9\, \mathbf{x}_{t-2}^T \mathbf{x}_{\tau-2} + \gamma_{10}\, \delta_{t,\tau}, \tag{3.21}$$

which generalizes the combination of RBF and linear kernels (see Equations 3.9 and 3.10) to the two preceding time steps. The term $\gamma_{10}\,\delta_{t,\tau}$ is a white-noise kernel component for modeling observation or process noise, with $\delta_{t,\tau}$ denoting the Kronecker delta. In the simpler first-order dynamics case ($z = 1$), the kernel excludes the $\mathbf{x}_{t-2}$-dependent terms.

Conceptually, this approach acts as a continuous-valued, non-linear extension of a hidden Markov model (HMM), where the state transitions are governed by a Gaussian process instead of discrete transition matrices. By integrating this GP dynamical prior with the GPLVM mapping from latent states to observed data, the GPDM captures both the manifold structure (through the GPLVM–now a component of the GPDM called the emission GP) and the temporal evolution (through the prior dynamics–the dynamical GP).

3.3.1 MAP Estimate with a Dynamical Prior

Just as in the GPLVM, the GPDM specifies a joint distribution over the observed data $\mathbf{Y}$ and the latent sequence $\mathbf{X} = \{\mathbf{x}_t\}_{t=1}^T$. We decompose this as [111, 112]:

$$p(\mathbf{Y}, \mathbf{X}) = p(\mathbf{Y} \mid \mathbf{X})\, p(\mathbf{X}), \tag{3.22}$$

where $p(\mathbf{Y} \mid \mathbf{X})$ is the standard GPLVM likelihood (mapping latent states to observations), and $p(\mathbf{X})$ is the dynamical prior. For the second-order case [111, 112],

$$p(\mathbf{X}) = p(\mathbf{x}_1)\, p(\mathbf{x}_2) \prod_{t=3}^{T} p\big(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_{t-2}\big), \tag{3.23}$$

while first-order dynamics ($z = 1$) uses only one previous state. The conditional $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \ldots)$ is specified by a GP with the corresponding autoregressive kernel.

To infer the latent trajectory $\mathbf{X}$ alongside the hyperparameters $\boldsymbol{\theta}$ (of both the dynamical and emission GPs), one can perform a MAP estimate as seen in equation 3.17. Alternatively, one can perform approximate Bayesian inference (e.g. variational methods [25, 26, 107] or MCMC [36, 63]) to capture uncertainty in $\mathbf{X}$ rather than simply finding a point estimate.

Compared to a basic GPLVM, the temporal prior $p(\mathbf{X})$ encourages smooth trajectories that reflect the true dynamical structure of the data. This additional constraint often leads to more coherent embeddings for time-series data, preventing discontinuities in the learned latent space.

3.4 Hierarchical GPLVMs

A *hierarchical* GPLVM extends the standard GPLVM by introducing multiple layers of latent variables, enabling the model to learn progressively more abstract representations of the data. In its simplest two-layer form, we have [23, 24]:

$$\mathbf{Y} = f^{(1)}(\mathbf{H}) + \boldsymbol{\epsilon}^{(1)}, \tag{3.24}$$

$$\mathbf{H} = f^{(2)}(\mathbf{X}) + \boldsymbol{\epsilon}^{(2)}, \tag{3.25}$$

where $\mathbf{Y}$ denotes the observed data, $\mathbf{H}$ is an intermediate latent variable, and $\mathbf{X}$ is a top-level latent variable. Each mapping function is given a Gaussian process prior [23, 24],

$$f^{(1)}(\mathbf{H}) \sim \mathcal{GP}\big(0,\, k^{(1)}(\mathbf{H}, \mathbf{H}')\big), \tag{3.26}$$

$$f^{(2)}(\mathbf{X}) \sim \mathcal{GP}\big(0,\, k^{(2)}(\mathbf{X}, \mathbf{X}')\big). \tag{3.27}$$

This hierarchical construction can be viewed as a deep GP, allowing more flexible, non-linear feature learning compared to a single-layer GPLVM. Advantages include:

1. Increased capacity to capture complex, non-linear structure in $\mathbf{Y}$.
2. Layers of latent variables that can disentangle different factors of variation.
3. A hierarchical covariance structure that may compactly represent multi-scale dependencies.

In principle, additional layers can be added, and each layer can incorporate different types of priors (e.g. dynamical priors for sequential data). In this thesis, we employ a GPDM only at the topmost layer for time-series modeling, while lower layers remain standard GP mappings.

3.4.1 Joint Likelihood and Conditional Integrations

Formally, a two-layer hierarchical GPLVM specifies a joint distribution over $\mathbf{Y}$, $\mathbf{H}$, and $\mathbf{X}$. The joint distribution can be factorized as [23, 24],

$$p(\mathbf{Y},\, \mathbf{H},\, \mathbf{X}) \;=\; p(\mathbf{Y} \mid \mathbf{H})\; p(\mathbf{H} \mid \mathbf{X})\; p(\mathbf{X}), \tag{3.28}$$

where

$$p(\mathbf{Y} \mid \mathbf{H}) = \int p(\mathbf{Y} \mid \mathbf{f}^{(1)})\, p(\mathbf{f}^{(1)} \mid \mathbf{H})\, d\mathbf{f}^{(1)}, \tag{3.29}$$

$$p(\mathbf{H} \mid \mathbf{X}) = \int p(\mathbf{H} \mid \mathbf{f}^{(2)})\, p(\mathbf{f}^{(2)} \mid \mathbf{X})\, d\mathbf{f}^{(2)}. \tag{3.30}$$

Here, $\mathbf{f}^{(1)}$ and $\mathbf{f}^{(2)}$ are vectors of function values related to equations 3.26 and 3.27, respectively. If we condition on the latent variables $\mathbf{H}$ or $\mathbf{X}$ being fixed, these Gaussian integrals over $\mathbf{f}^{(1)}$ or $\mathbf{f}^{(2)}$ are tractable. In other words, given $\mathbf{H}$, the first layer reduces to a standard GP regression, and similarly for the second layer given $\mathbf{X}$.

3.4.2 Maximum Likelihood and MAP Estimation

As with the standard GPLVM, one faces an intractable integral if attempting to integrate out $\mathbf{H}$ and $\mathbf{X}$ [23, 24]:

$$p(\mathbf{Y}) \;=\; \int p(\mathbf{Y} \mid \mathbf{H})\, p(\mathbf{H} \mid \mathbf{X})\, p(\mathbf{X})\, d\mathbf{H}\, d\mathbf{X}.$$

Hence, we treat both $\mathbf{H}$ and $\mathbf{X}$ as learnable parameters and maximize [23, 24],

$$L = \log p(\mathbf{Y} \mid \mathbf{H}) \;+\; \log p(\mathbf{H} \mid \mathbf{X}) \;+\; \log p(\mathbf{X}),$$

plus additional structured priors or regularization terms on $\mathbf{H}, \mathbf{X}$.

3.4.3 Illustrative Two-Layer Example

For concreteness, if one treats $\mathbf{H}$ and $\mathbf{X}$ as fixed parameters and the noise as Gaussian, each layer's conditional log-likelihood resembles that of a standard GP. We can write a simplified joint log-likelihood as [23, 24]

$$\begin{aligned} L = &-\frac{D_y}{2}\log\left|\mathbf{K}^{(1)}\right| - \frac{1}{2}\operatorname{tr}\left(\left(\mathbf{K}^{(1)}\right)^{-1}\mathbf{Y}\mathbf{Y}^\top\right) \\ &-\frac{D_h}{2}\log\left|\mathbf{K}^{(2)}\right| - \frac{1}{2}\operatorname{tr}\left(\left(\mathbf{K}^{(2)}\right)^{-1}\mathbf{H}\mathbf{H}^\top\right) - C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{H}), \end{aligned} \tag{3.31}$$

where $\mathbf{K}^{(1)}$ is the kernel matrix at the first layer (built from $\mathbf{H}$), $\mathbf{K}^{(2)}$ is the kernel at the second layer (built from $\mathbf{X}$), and D_y and D_h denote the dimensionalities of $\mathbf{Y}$ and $\mathbf{H}$, respectively. The term $C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{H})$ collects additional contributions such as normalization constants, noise terms, or latent priors, depending on the exact model specification. Because $\mathbf{H}$ and $\mathbf{X}$ enter nonlinearly through the kernels, optimizing L with respect to these variables typically requires gradient-based or approximate inference techniques.

In summary, hierarchical GPLVMs (sometimes called deep GPs) facilitate richer, multi-level latent representations at the cost of increased complexity. For time-series data, one can further augment the top-level latent space $\mathbf{X}$ with a GPDM, thereby combining both temporal dynamics and hierarchical feature extraction [103, 104].

3.5 Sparse GP Approximations

A key challenge for GPs is their cubic $\mathcal{O}(N^3)$ computational cost in the number of training points N [83, 114]. To mitigate this, *sparse approximation* methods introduce a smaller set of M *inducing points*, where typically $M \ll N$. These inducing points summarize information from the full dataset, yielding approximate GP posteriors with reduced complexity, decreasing computational requirements from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$.

Constructing the Approximate Model.

Bui et al. [12] presents a unifying framework for understanding and deriving sparse GP approximations. Let $\mathbf{f} = [f_1, \ldots, f_N]^T$ denote function values at N inputs $\mathbf{X}$, which in standard GPs represents training data rather than the matrix of latent variables used in the previous three sections. Let $\mathbf{u} = [u_1, \ldots, u_M]^T$ denote the function values at M inducing inputs $\mathbf{Z}$. The joint prior factorizes as [12],

$$p(\mathbf{f}, \mathbf{u} \mid \boldsymbol{\theta}) = p(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})\, p(\mathbf{u} \mid \boldsymbol{\theta}), \tag{3.32}$$

where $p(\mathbf{u} \mid \boldsymbol{\theta})$ is the Gaussian prior on the inducing points, and $p(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$ describes how the remaining function values correlate with the inducing points. Although this exact model leads to $\mathcal{O}(N^3)$ complexity, an approximate joint distribution with a simpler conditional structure can instead be introduced that still involves $\mathbf{u}$ but reduces the dependencies among the N data points [12]:

$$q(\mathbf{f}, \mathbf{u} \mid \boldsymbol{\theta}) = p(\mathbf{u} \mid \boldsymbol{\theta})\, q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta}). \tag{3.33}$$

The difference lies in the conditional $q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$, whose form is chosen to yield matrix factorizations or low-rank structures that are easier to compute. In essence, $q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$ replaces the original Gaussian conditional $p(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$ by a parameterized family (e.g., diagonal corrections, block-diagonal structures, or entirely deterministic mappings). This approach recasts GP approximation as simply performing inference in a "new" generative model defined by equation 3.33 together with the original data likelihood [12]:

$$q(\mathbf{y}, \mathbf{f}, \mathbf{u} \mid \boldsymbol{\theta}) = q(\mathbf{f}, \mathbf{u} \mid \boldsymbol{\theta}) \prod_{n=1}^{N} p(y_n \mid f_n). \tag{3.34}$$

Since $p(\mathbf{u} \mid \boldsymbol{\theta})$ remains a standard Gaussian prior on the inducing points, one can still rely on the usual GP machinery, but now all inferences are governed by the modified conditional $q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$.

Unifying Common Approximations.

Different choices for $q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta})$ subsume well-known sparse GP methods such as the deterministic training conditional (DTC), fully independent training conditional (FITC), and partially independent training conditional (PITC) [83]. For instance, one may write [12]:

$$q(\mathbf{f} \mid \mathbf{u}, \boldsymbol{\theta}) \;=\; \prod_{b=1}^{B} \mathcal{N}\Big(\mathbf{f}_b;\; \mathbf{K}_{\mathbf{f}_b\mathbf{u}}\, \mathbf{K}_{\mathbf{uu}}^{-1}\, \mathbf{u},\; \alpha\, \mathbf{D}_{\mathbf{f}_b\mathbf{f}_b}\Big), \tag{3.35}$$

where $\mathbf{f}$ has been partitioned into B subsets (or blocks) $\{\mathbf{f}_b\}$, and $\mathbf{D}_{\mathbf{f}_b\mathbf{f}_b}$ is some diagonal or block-diagonal correction to retain partial dependence. Setting $\alpha = 1$ with $B = N$ and extracting *only* the diagonal elements of $\mathbf{K_{ff}} - \mathbf{K_{fu}}\mathbf{K_{uu}^{-1}}\mathbf{K_{uf}}$ recovers the FITC approximation (presented below). If instead $\alpha \to 0$, one obtains the DTC approximation, while PITC uses $\alpha = 1$. Thus, equation 3.35 succinctly encodes a spectrum of sparse approaches, each introducing inducing inputs $\mathbf{Z}$ to reduce the full GP prior's rank [12].

3.5.1 FITC Approximation

This section outlines the *fully independent training conditional* (FITC) approximation [83, 93] as a concrete realization of the general framework presented above in section 3.5. FITC provides a computationally efficient way to approximate the conditional distribution $p(\mathbf{f} \mid \mathbf{u})$ in a GP. Recall from section 3.5 that $\mathbf{f} = [f_1, \ldots, f_N]^T$ denotes latent function values at the N training inputs, and $\mathbf{u} = [u_1, \ldots, u_M]^T$ denotes the function values at M inducing inputs (with $M \ll N$). FITC replaces the exact Gaussian conditional with [83],

$$p(\mathbf{f} \mid \mathbf{u}) \;\approx\; q(\mathbf{f} \mid \mathbf{u}) \;=\; \mathcal{N}\Big(\mathbf{K_{fu}}\, \mathbf{K_{uu}^{-1}}\, \mathbf{u},\; \mathrm{diag}\big[\mathbf{K_{ff}} - \mathbf{Q_{ff}}\big]\Big), \tag{3.36}$$

where $\mathbf{Q_{ab}} = \mathbf{K_{au}}\, \mathbf{K_{uu}^{-1}}\, \mathbf{K_{ub}}$ and $\mathrm{diag}[\cdot]$ retains only the diagonal elements of a matrix. Here, $\mathbf{K_{fu}}$ is the $(N \times M)$ cross-covariance between N training inputs and M inducing inputs, $\mathbf{K_{uu}}$ is the $(M \times M)$ kernel matrix among the inducing inputs, and $\mathbf{K_{ff}}$ is the $(N \times N)$ covariance among the training inputs.

Approximate Marginal Likelihood.

Under equation 3.36, the approximate marginal likelihood of the observed outputs $\mathbf{y}$ is given by [83],

$$p(\mathbf{y}) \;\approx\; q(\mathbf{y}) \;=\; \mathcal{N}\Big(\mathbf{0},\; \mathbf{Q_{ff}} \;-\; \mathrm{diag}\big[\mathbf{K_{ff}} - \mathbf{Q_{ff}}\big] \;+\; \sigma^2\mathbf{I}\Big). \tag{3.37}$$

The diagonal term $\mathrm{diag}[\mathbf{K_{ff}} - \mathbf{Q_{ff}}]$ is often called the *diagonal correction*, and it preserves each training point's marginal variance while discarding off-diagonal correlations between them. The log of equation 3.37 then takes the usual Gaussian form [83, 93],

$$L \;=\; -\tfrac{1}{2}\,(\mathbf{y}^T\mathbf{A}^{-1}\mathbf{y}) \;-\; \tfrac{1}{2}\,\log\lvert\mathbf{A}\rvert \;-\; \tfrac{N}{2}\,\log(2\pi), \tag{3.38}$$

where $\mathbf{A} \;=\; \mathbf{Q_{ff}} \;+\; \mathrm{diag}\big[\mathbf{K_{ff}} - \mathbf{Q_{ff}}\big] \;+\; \sigma^2\mathbf{I}$. Although $\mathbf{A}$ is still an $(N \times N)$ matrix, it inherits low-rank plus diagonal structure that can be exploited for $\mathcal{O}(NM^2)$ computations.

Predictive Distribution.

Another key advantage of sparse GP approximations becomes evident at prediction time. While training complexity scales as $\mathcal{O}(NM^2)$, making predictions at new test points requires only $\mathcal{O}(M)$ computational

operations. These operations involve matrix calculations with the inducing points rather than the full training set. Specifically, to predict at new inputs $\mathbf{X}_*$, the predictive distribution of $\mathbf{f}_*$ is Gaussian with mean $\boldsymbol{\mu}_*$ and covariance $\boldsymbol{\Sigma}_*$ [83]:

$$p(\mathbf{f}_*|\mathbf{X}_*, \mathbf{Y}, \mathbf{X}) = \mathcal{N}(\mathbf{f}_*; \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*), \tag{3.39}$$

where:

$$\boldsymbol{\mu}_* = \mathbf{K}_{*\mathbf{u}}\boldsymbol{\Sigma}\mathbf{K}_{\mathbf{uf}}\boldsymbol{\Lambda}^{-1}\mathbf{Y}, \quad \boldsymbol{\Sigma}_* = \mathbf{K}_{**} + \mathbf{K}_{*\mathbf{u}}\boldsymbol{\Sigma}\mathbf{K}_{\mathbf{u}*} - \mathbf{Q}_{**}, \tag{3.40}$$

where $\boldsymbol{\Lambda} = \text{diag}[\mathbf{K}_{\mathbf{ff}} - \mathbf{Q}_{\mathbf{ff}}] + \sigma^2_{noise}\mathbf{I}$, and $\boldsymbol{\Sigma} = (\mathbf{K}_{\mathbf{uu}} + \mathbf{K}_{\mathbf{uf}}\boldsymbol{\Lambda}^{-1}\mathbf{K}_{\mathbf{fu}})^{-1}$ combines various kernel terms. $\mathbf{K}_{*\mathbf{u}}$ denotes the covariance between test points and inducing points, while $\mathbf{K}_{**}$ is the covariance between test points. Further details about the FITC approximation can be found in Quinonero-Candela and Rasmussen [83]. DTC, also employed in this work, has the same basic formulation where $\alpha \to 0$ in equation 3.35, simplifying the derivation.

3.5.2 Sparse Approximations in GPLVMs and GPDMs

Recall that in the basic GPLVM, a Gaussian process prior is placed over the mapping from the latent space $\mathbf{X} \in \mathbb{R}^{N\times Q}$ (with Q-dimensional latent points) to the observed data space $\mathbf{Y} \in \mathbb{R}^{N\times D}$. A GPDM extends the GPLVM to time-series or sequential data. It posits a latent state sequence $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ evolving according to a GP *dynamical* mapping, combined with a GP *emission* mapping from latent states to observed data $\{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ (alone, the GP emission model is equivalent to a GPLVM). Formally [111, 112],

$$p(\mathbf{Y}, \mathbf{X} \mid \boldsymbol{\theta}) = \underbrace{\prod_{t=2}^{N} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \boldsymbol{\theta}_{\text{dyn}})}_{\text{GP dynamics}} \times \underbrace{\prod_{t=1}^{N} p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_{\text{emit}})}_{\text{GP emission}}, \tag{3.41}$$

where each factor is a Gaussian process in its respective function space. However, the same $\mathcal{O}(N^3)$ complexity pitfalls arise when forming each covariance over N time steps.

Inducing Points for Transitions and Emissions.

To alleviate the cost, $M \ll N$ inducing points are introduced for both the dynamical and emission functions, applying sparse approximations in a manner analogous to section 3.5. In particular, the following replacements are made [64]:

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \longrightarrow q(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{\text{dyn}}), \quad p(\mathbf{y}_t \mid \mathbf{x}_t) \longrightarrow q(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{u}_{\text{emit}}), \tag{3.42}$$

where $\mathbf{u}_{\text{dyn}}$ and $\mathbf{u}_{\text{emit}}$ collect the inducing variables for the two GP components. As with FITC (section 3.5.1), a factorized conditional for $q(\cdot|\mathbf{u})$ plus diagonal or block-diagonal corrections may be chosen.

3.5.3 FITC Approximation in GPLVMs

When FITC is applied to GPLVMs, the sparse approximation framework is adapted directly to the mapping from latent variables to observations. Building on the approach established in section 3.5.1, the conditional distribution in a latent variable model is stated as [64]:

$$q(\mathbf{Y}|\mathbf{X}, \mathbf{u}) = \mathcal{N}\big(\mathbf{K}_{\mathbf{fu}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{u}, \text{diag}[\mathbf{K}_{\mathbf{ff}} - \mathbf{Q}_{\mathbf{ff}}] + \sigma^2\mathbf{I}\big), \tag{3.43}$$

where $\mathbf{X}$ represents the latent variables, and σ^2 is the noise variance. The key characteristic of FITC in this context is the diagonal treatment of the correction term, which enforces independence between data points conditional on the inducing variables $\mathbf{u}$.

Reintroducing the prior over the inducing variables, the marginal log-likelihood for the GPLVM becomes [64]:

$$L = -\frac{D}{2}\log(2\pi) - \frac{D}{2}\log|\mathbf{A}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^T\mathbf{A}^{-1}\right), \tag{3.44}$$

where $\mathbf{A} = \mathbf{Q}_{\mathbf{ff}} + \mathrm{diag}[\mathbf{K}_{\mathbf{ff}} - \mathbf{Q}_{\mathbf{ff}}] + \sigma^2\mathbf{I}$ and D is the dimensionality of the observed data. This log-likelihood exhibits the same structure as in standard FITC but is adapted for the GPLVM context where we optimize both the latent variables $\mathbf{X}$ and inducing inputs $\mathbf{Z}$ simultaneously. The computational complexity remains $\mathcal{O}(NM^2)$, making it feasible to train GPLVMs on larger data sets.

3.6 GPLVM Back-Constraints

Back-constraints in GPLVMs enforce specific topological constraints on the latent space by explicitly defining a mapping from the data space to the latent space [108]. This mapping ensures that points close in the data space remain close in the latent space. Formally, we can express the back-constraint mapping as:

$$\mathbf{x}_i = g(\mathbf{y}_i, \mathbf{W}) \tag{3.45}$$

where $g(\cdot)$ is the back-constraint function and $\mathbf{W}$ represents the parameters of this mapping. During optimization, instead of directly optimizing the latent points $\mathbf{X}$, we optimize the parameters $\mathbf{W}$ of the back-constraint function.

3.7 Mixture-of-Experts Models

Mixture-of-experts models, originally proposed by Jacobs et al. [49], provide a framework for decomposing complex tasks into regions handled by specialized experts. This approach is particularly valuable for movement modeling, where different types of movements require mastering distinct control strategies.

The model takes the form:

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\alpha\in\mathcal{A}} p(\alpha|\mathbf{x})p(\mathbf{y}|\mathbf{x},\alpha) \tag{3.46}$$

where $\alpha \in \mathcal{A}$ indexes the experts, $p(\alpha|\mathbf{x})$ is the gating function determining each expert's responsibility, and $p(\mathbf{y}|\mathbf{x},\alpha)$ is the expert-specific output distribution.

The system learns both expert specializations and the gating function through optimization. For movement data, this often involves learning a shared state representation, developing specialized dynamics models, and establishing smooth transitions between expert regions.

Key advantages include accurate modeling of diverse movement types, natural movement classification, interpolation between movement types with overlapping expert responsibilities, and capturing the hierarchical nature of motor control under different task demands.

4. Data Sets

This work utilizes three distinct data sets of human movement. Two of these data sets were collected for this research from specifically designed experiments: a reach-to-grasp experiment focusing on upper limb movements toward fixed targets, and a bimanual experiment investigating coordinated two-handed actions in activities of daily living. The third data set was obtained from the Carnegie Mellon University Motion Capture Database (CMU Graphics Lab) [21], which provides open-access motion capture recordings for research purposes.

4.1 Reach-to-Grasp Experiment

4.1.1 Experimental Design

The reach-to-grasp experiment was designed to capture coordinated arm and hand movements during targeted reaching actions. Infrared (IR) motion capture and electromyographic (EMG) measurements were synchronized and recorded simultaneously.

Targets for grasping were arranged on an experimental apparatus consisted of nine rectangular wooden blocks positioned in a 3×3 matrix, with uniform spacing of 25 cm both vertically and horizontally. This configuration allowed for systematic investigation of reaching movements across different spatial locations while maintaining controlled experimental conditions.

The participant was seated facing the apparatus with an erect posture, their right shoulder approximately aligned with the apparatus midline. This positioning minimized trunk movement during reaching. Each trial consisted of nine reach-to-grasp movements, one for each target block, with a randomized order. The protocol followed this sequence:

1. Starting position: Right hand palm-down on a hip-level platform
2. Reach and grasp (3-second duration)
3. Hold position (3-second duration)
4. Return to starting position (3-second duration)
5. Rest period (3-second duration)

Movement timing was regulated using a 60 bps metronome.

4.1.2 Data Collection Systems

Subjects

One healthy, left-handed, adult male (age: 39 years, height: 178 cm, weight: 82 kg) with no prior history of neuromuscular disorders participated in the study.

Motion Capture Setup

Movement kinematics were recorded using a VICON motion capture system comprising eight infrared cameras operating at 256.41 Hz (VICON Nexus version 2.8) [100]. A total of 48 infrared markers were employed:

- 22 markers (4 mm) covering the right hand and wrist
- 3 markers (4 mm) on the left wrist and hand
- 23 markers (14 mm) distributed across arms, shoulders, trunk, pelvis, and head

Marker placement prioritized stable anatomical locations to minimize artifact from soft tissue movement during recording.

Electromyographic Recording

Surface electromyography (sEMG) was recorded using both high-density arrays and bipolar electrodes:

- Three 8×8 high-density surface electrode arrays (OT Bioelettronica) placed circumferentially around the forearm
- 16 bipolar surface electrodes on larger muscles of the upper arm, shoulder, chest, and back

EMG signals were recorded using an OT Bioelettronica Quattrocento 400-channel amplifier with the following parameters:

- Sampling rate: 2,848 Hz
- Amplification gain: 150 V/V
- Online bandpass filter: 10-900 Hz

4.1.3 Data Processing

EMG Processing

Raw EMG signals underwent the following processing steps:

1. Digital bandpass filtering (20-450 Hz, fourth-order Butterworth)
2. Powerline interference removal (49-51 Hz, second-order Butterworth band-stop)
3. Full-wave rectification
4. Linear envelope extraction (10 Hz, second-order Butterworth low-pass)

To reduce redundancy in the HD-sEMG signals, each array was divided into quadrants and averaged, resulting in 12 consolidated channels representing forearm muscle activity.

Motion Capture Processing

Kinematic data processing included:

1. Marker labeling and gap-filling
2. Kinematic skeleton creation in Biovision hierarchical format

Temporal Alignment

All data were segmented into individual reach-to-grasp movements with the following considerations:

- Movement segments extended 200 ms beyond the primary movement interval
- Data were normalized to 90 timepoints using linear interpolation
- EMG and kinematic data were synchronized during recording

4.2 Bimanual Experiment

4.2.1 Experimental Design

The bimanual experiment was designed to investigate coordinated two-handed movements during activities of daily living (ADLs). Infrared (IR) and inertial measurement unit (IMU) motion capture, and electromyographic (EMG) measurements were synchronized and recorded simultaneously.

Five distinct tasks were selected to represent common bimanual activities. Each task was varied across one of more additional conditions. Each subject performed 10 trials of each task for a single condition, giving a total of 140 trials per subject. All trials began and ended with participants in a neutral position, defined as both hands placed palm-down on the desk. The tasks are enumerated followed by their bulleted conditions below:

1. Lifting a basket (vertical displacement)
    - Full-height (about level with the head) and half-height (about level with the sternum) conditions
    - With and without a 2kg weight placed in the basket

2. Lifting a basket approximately 10cm and moving it horizontally across a table (lateral displacement)
   - Left-to-right and right-to-left directions
   - With and without a 2kg weight placed in the basket
3. Rotating a basket 90 degrees
   - Clockwise and counterclockwise
4. Opening and closing a jar
   - Dominant and non-dominant hand
5. Cutting with a knife
   - Dominant and non-dominant hand

4.2.2 Subjects

Five healthy subjects between the age of 23 and 28 years with no prior history of neuromuscular disorders participated in the full experimental protocol, including IR and IMU motion capture, and EMG recordings.

4.2.3 Data Collection Systems

IR Motion Capture

Movement kinematics were recorded using a VICON motion capture system (Shogun) [101] comprising eight infrared cameras operating at 256.41 Hz. A total of 55 infrared markers were employed. The marker set included:

- Head: 5 markers
- Shoulders: 4 markers (2 per side)
- Chest: 2 markers
- Back: 2 markers
- Waist: 6 markers (sides)
- Upper arms: 4 markers (2 per side)
- Lower arms: 4 markers (2 per side)
- Wrists: 4 markers (2 per side)
- Back of hands: 4 markers (2 per side)
- Fingers: 20 markers (2 per finger)
  - First marker immediately distal to the last joint of each finger
  - Index, ring, and pinky fingers: Second marker immediately distal to the second joint
  - Thumbs and middle fingers: Second marker immediately distal to the first joint

IMU Motion Capture

IMU data was collected using an Xsens MVN system [86] comprising 17 IMUs. The goal of the IMU recordings were to enable the integration of a portable motion capture system with the higher-resolution, more articulated IR motion capture system. The IR motion capture was intended for training the motion synthesis model with a strong foundation and prior on movement and finger articulation, while the IMU data (alongside a simplified EMG system) was intended for inference during real-time motion prediction and control applications. However, the project ultimately diverged from these pursuits.

17 IMUs were placed according to the standard Xsens MVN system protocol, including the trunk, upper and lower limbs, but excluding the fingers. Lower body was including for alignment with the Xsens MVN software, but was not relevant for the set of tasks. Specific placement of the IMUs can be found by reviewing the Xsens MVN protocol, but the general locations are as follows: head, shoulders, sternum, pelvis, upper and lower arms, upper and lower legs, hands, and feet.

Electromyography

EMG signals were recorded using an OT Bioelettronica Quattrocento 400-channel amplifier with the following parameters:

- Sampling rate: 2,848 Hz
- Amplification gain: 150 V/V
- Online bandpass filter: 10-900 Hz

Placement of EMG electrodes was as follows:

- 12 bipolar electrodes placed bilaterally on the upper arms and shoulders:
  - Biceps long head
  - Triceps brachii lateral head
  - Triceps brachii medial head
  - Anterior deltoid
  - Posterior deltoid
  - Upper trapezius
- Four 4x4 HD electrode arrays total, with one placed on either side of both forearms
- Reference electrodes on the medial epicondyle of the humerus bone and the acromion process of the scapula

4.2.4 Data Processing

Note on Selective Processing

While data was collected using three measurement systems (IR and IMU motion capture, and EMG), our subsequent analysis of the reach-to-grasp experiment results led to a strategic focus solely on the IR motion capture data for studies on the bimanual data set.

Motion Capture Processing

Motion capture data processing followed the same general procedure as the reach-to-grasp experiment:

1. Marker labeling and gap-filling using Vicon Shogun
2. Kinematic skeleton creation in Biovision hierarchical format (MotionBuilder 2020)
3. All movements were aligned to a universal starting position

Temporal Alignment

All data were segmented into individual reach-to-grasp movements with the following considerations:

- Movement segments extended 200 ms beyond the primary movement interval
- EMG and kinematic data were synchronized during recording
- Data were normalized to 100 timepoints using linear interpolation

4.3 CMU Motion Capture Dataset

4.3.1 Dataset Overview

The Carnegie Mellon University Motion Capture Database provides open-access motion capture recordings for research purposes [21]. From this extensive database, we selected a subset of recordings focusing on distinct full-body movements:

- Bending down
- Soccer ball kicking
- Lateral jumping
- Forward kicking
- Lunging
- Punching
- Breaststroke
- Butterfly stroke

For each movement category, six complete trials were selected. For cyclic movements like swimming forms, a single complete cycle was extracted from each trial.

4.3.2 Recording System

The CMU motion capture laboratory utilized:

- 12 Vicon infrared MX-40 cameras
- Recording frequency: 120 Hz
- Camera resolution: 4 megapixels
- Capture volume: approximately 3m × 8m

4.3.3 Marker Setup

Subjects wore 41 markers placed on a black capture suit. The marker set provided joint angle data for 77 features representing:

- Neck joints
- Spinal segments
- Pelvic joints
- Arm joints (shoulder, elbow, wrist)
- Leg joints (hip, knee, ankle)

Finger and toe movements were not captured in these recordings.

Acknowledgment

The creation of the CMU motion capture database was supported by NSF Grant #0196217.

Part III.

Projects

5. Preface

This section presents the research projects I conducted during my PhD, arranged chronologically from my first publication. Each project is preceded by an interlude chapter that provides context and motivation. While preserving many sections of the original papers verbatim, I have streamlined them for conciseness and relevance within the broader thesis context by removing redundant passages from introductions, background sections, and methods. In some instances, I have added new content to elaborate on details that were omitted from the original papers due to space constraints. Supplementary material for the third paper is included in the appendix (A).

6. Interlude 1: Converging on GPDMs

My first project emerged from a collaboration on the KONSENS-NHE initiative, funded by the Baden-Württemberg Stiftung GmbH. The project aimed to develop a context-sensitive, neurally controlled hand exoskeleton to restore everyday functionality and autonomy for individuals recovering from brain and spinal cord injuries. This collaborative effort involved my colleagues Prerana Kumar, who contributed significantly to data collection and EMG pre-processing, and Nick Taubert, who led the model design, code development, and manuscript writing. We were supervised by my primary mentor and PI, Prof. Dr. Martin Giese, and Dr. Leonardo Gizzi, who directed the data collection efforts. My contributions focused on experimental design, data collection, motion capture data pre-processing, model design, and authoring the manuscript's background section on data collection methodology, as well as contributing to overall manuscript revisions.

Our primary objective was to develop a sophisticated movement model capable of predicting finger and wrist articulation from the combined input of EMG signals and motion capture data. This set of data was collected from our reach-to-grasp experiment described in section 4.1. Initially, Nick and I pursued different modeling approaches. Building on his prior expertise, Nick developed GPDMs, whereas I focused on movement primitives models using the FADA toolbox (Fourier-based Anechoic Demixing Algorithm) [18].

Through some trial-and-error and a lot of testing, we determined that GPDMs were better aligned with our research objectives. As discussed in the Introduction section 2.6, GPDMs offer several advantages practical in clinical settings: they provide interpretable results, quantify prediction uncertainty, enable real-time inference, and achieve high accuracy with limited training data. Building upon Nick's model, we co-authored the paper *Reactive Hand Movements from EMG Signals based on Hierarchical Gaussian Process Dynamical Models* [104], which established the foundation for my subsequent PhD research. An abridged version of this paper is presented in the next section (7), with most of the text reproduced verbatim from the original manuscript.

7. Reactive Hand Movements from EMG Signals based on Hierarchical Gaussian Process Dynamical Models

7.1 Abstract

The prediction of finger kinematics from EMG signals is a difficult problem due to the high level of noise in recorded biological signals. In order to improve the quality of such predictions, we propose a Bayesian inference architecture that enables the combination of multiple sources of sensory information with an accurate and flexible model for the online prediction of high-dimensional kinematics. Our method integrates hierarchical Gaussian process latent variable models (GP-LVMs) for nonlinear dimension reduction with Gaussian process dynamical models (GPDMs) to represent movement dynamics in latent space. Using several additional approximations, we make the resulting sophisticated inference architecture real-time capable. Our results demonstrate that the prediction of hand kinematics can be substantially improved by inclusion of information from the online-measured arm kinematics, and by exploiting learned online generative models of finger kinematics. The proposed architecture provides a highly flexible framework for the integration of accurate generative models with high-dimensional motion in real-time inference and control problems.

7.2 Introduction

The prediction of finger kinematics from EMG signals presents significant challenges due to the high dimensionality of human motion data and the inherent noise in biological signals. While dynamical systems approaches have shown promise in modeling coordinated motor patterns [41], their application to prosthesis control remains challenging due to the limited dimensionality and noise characteristics of available control signals.

To address these challenges, we propose a hierarchical probabilistic graphical model [6] that integrates neural signal decoding with motion synthesis. Our approach combines Gaussian process latent variable models (GP-LVMs) [62] with Gaussian process dynamical models (GPDMs) [112] in a real-time capable architecture. This framework enables the estimation of hand kinematics from EMG signals while incorporating additional kinematic data from arm movements. The resulting model provides a foundation for controlling hand orthoses in patients who retain arm function but experience impaired grasping abilities, such as in post-stroke rehabilitation [10].

7.3 Background

The industry standards and the typical benchmarks in research for online hand and arm myocontrol comprise the two-electrode "conventional control" approaches, co-contraction control (CC) and slope control (SC) [43]. Both require users to make targeted muscle contractions for switching between two single degrees of freedom (DOF) modes of control—operating a wrist rotation and opening/closing the hand. Users often consider these systems to be unintuitive compared to natural hand and arm function, difficult to use in everyday life, and to be fatiguing [35]. Other approaches exploiting classification and multiple DOFs regression methods have been shown to produce better results with the same hardware [43].

More advanced techniques have effectively employed methods of auto-encoding for dimensionality reduction [110], have utilized biomechanical models for imposing realistic system constraints [34], or have incorporated proprioceptive feedback or direct cortical feedback into sensory-motor brain areas [37]. Here, we explore the use of generative hierarchical Gaussian Process models for establishing control by non-invasive EMG signals, augmented by predictive signals derived from the measurement of additional arm kinematic signals.

To achieve real-time performance in our hierarchical model, we introduce a fast approximation method for inference by learning an approximate inverse of our generative model with GP, which we refer to as back-projection. This approach to making inference tractable was inspired by the Helmholtz Machine [27], which also learns an explicit inversion of a generative model.

7.4 Model components

Our model architecture is shown in Figure 7.1a. It is composed of a hierarchical combination of GPLVMs, and a top level that is implemented by a GPDM-like dynamical layer. See Background & Methods sections 3.2 and 3.3 for an introduction to GPLVMs and GPDMs.

7.4.1 Sparse aproximation

For a general overview on the sparse approximation of GPs, see the Background & Methods sections starting with 3.5. Below we specify the sparse approximation used in this application, taken directly from the original paper [104].

The $O(N^3)$ complexity of the model was solved via sparse approximation. For this purpose, the noise free function value set $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_N]^T$, where $\mathrm{f}_d = f_d(\mathbf{x})$ with $\mathbf{f} = [\mathrm{f}_1, \ldots, \mathrm{f}_D]^T$, is approximated by selection of

a small set of $M \ll N$ pseudo-points $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]^\mathrm{T}$. It is assumed that training and test points of the Gaussian Process are approximately conditionally independent, if conditioned on their pseudo-points. Under this assumption, the GP prior can be approximated as follows:

$$q(\mathbf{f}|\boldsymbol{\theta}) = \int p(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta}) p(\mathbf{u}|\boldsymbol{\theta}) \mathrm{d}\mathbf{u}, \tag{7.1}$$

where $\boldsymbol{\theta}$ are the model hyperparameters and $p(\mathbf{u}|\boldsymbol{\theta})$ is the GP prior over $\mathbf{u}$. The conditional relationship between $\mathbf{u}$ and $\mathbf{f}$ is fundamental for this equation. With the GP priors $p(\mathbf{f}|\boldsymbol{\theta})$ and $p(\mathbf{u}|\boldsymbol{\theta})$ it is possible to derive from their joint probability the conditional dependency. This leads to the result $p(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}; \mathbf{K_{fu}}\mathbf{K_{uu}^{-1}}\mathbf{u}, \mathbf{D_{ff}})$ where $\mathbf{D_{ff}} = \mathbf{K_{ff}} - \mathbf{Q_{ff}}$ and $\mathbf{Q_{ff}} = \mathbf{K_{fu}}\mathbf{K_{uu}^{-1}}\mathbf{K_{uf}}$. The constructed matrices correspond to the covariance function evaluations at the pseudo-point input locations $\{[\mathbf{x_u}]_m\}_{m=1}^M$, i.e. $[\mathbf{K_{uu}}]_{m,m'} = k_\mathbf{f}([\mathbf{x_u}]_m, [\mathbf{x_u}]_{m'})$ and similarly covariance function evaluations between pseudo-point input and data locations $[\mathbf{K_{fu}}]_{n,m} = k_\mathbf{f}(\mathbf{x}_n, [\mathbf{x_u}]_m)$. Unfortunately, even this equation it is still cubic in terms of its computational complexity. However, the matrices of $p(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta})$ can be approximated by simpler forms, resulting in computationally tractable distributions with $q(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta}) \approx p(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta})$ [13]. We chose a subset of pseudo-inputs which parameterize the GP prior through the Deterministic Training Conditional (DTC) approximation [64] in order to ensure real-time capability for prediction and to render our approach feasible for larger data sets,

$$q(\mathbf{f}|\mathbf{u}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}|\mathbf{K_{fu}}\mathbf{K_{uu}^{-1}}\mathbf{u}, \mathbf{0}). \tag{7.2}$$

This approximation assumes that $f(\mathbf{x})$ is fully determined by the pseudo-point inputs and reduces the computational cost to $O(M^2N)$ during learning. The prior $q(\mathbf{f}|\boldsymbol{\theta})$ with DTC approximation can be combined with the likelihood $p(\mathbf{y}_d|\mathbf{f}_d, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}_d|\mathbf{f}_d, \gamma_4^{-1}\mathbf{I})$:

$$p(\mathbf{Y}|\theta) = \prod_{d=1}^{D} \int p(\mathbf{y}_{:,d}|\mathbf{f}_{:,d}, \boldsymbol{\theta}) q(\mathbf{f}_{:,d}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{f}_d \tag{7.3}$$

$$= \prod_{d=1}^{D} \mathcal{N}(\mathbf{y}_{:,d}|\mathbf{0}, \mathbf{K_{fu}}\mathbf{K_{uu}^{-1}}\mathbf{K_{uf}} + \gamma_4^{-1}\mathbf{I}). \tag{7.4}$$

where γ_4 is the noise precision into which any Gaussian observation noise can be absorbed and $\mathbf{y}_{:,d}, \mathbf{f}_{:,d}$ are the vectors for the dth dimension of $\mathbf{Y}$ and $\mathbf{F}$. Combined with priors $p(\mathbf{X})$ and $p(\theta)$ we got the log-marginal, reformulated with matrix inversion lemma [64] for $O(M^2N)$:

$$\mathcal{L} = -\frac{D(N-M)}{2} \ln 2\pi - \frac{\gamma_4}{2} tr(\mathbf{Y}\mathbf{Y}^\mathrm{T}) \frac{D}{2} \ln |\mathbf{K_{uu}^{-1}}| + \frac{D}{2} \ln |\mathbf{A}| \tag{7.5}$$

$$-\frac{\gamma_4}{2} tr(\mathbf{A^{-1}}\mathbf{K_{uf}}\mathbf{Y}\mathbf{Y}^\mathrm{T}\mathbf{K_{fu}}) - \frac{1}{2}\sum_{n=1}^{N} \mathbf{x}_n^\mathrm{T}\mathbf{x}_n - \sum_j \ln \gamma_j, \tag{7.6}$$

with $\mathbf{A} = \gamma_4^{-1}\mathbf{K_{uu}} - \mathbf{K_{uf}}\mathbf{K_{fu}}$. This implies that the computation time depends only on the number of pseudo- and test inputs, which enables real-time performance.

For the prior function of new test data $p(\tilde{\mathbf{Y}}|\theta)$, we followed the same derivations as in equations (7.1)-(7.4), replacing $\mathbf{f}$ with test function values $\tilde{\mathbf{f}}$, assumed conditional independence between $\tilde{\mathbf{f}}$ and $\mathbf{f}$. With the joint probability of the new GP priors $p(\mathbf{Y}|\theta)$ and $p(\tilde{\mathbf{Y}}|\theta)$, a posterior distribution $p(\tilde{\mathbf{y}}|\mathbf{Y}, \mathbf{X}, \mathbf{x}_*, \theta)$ for the prediction of new test outputs $\tilde{\mathbf{y}}$ with given test inputs $\mathbf{x}_*$ can be derived [65],

$$\tilde{\mathbf{y}} \sim \mathcal{N}(\mu(\mathbf{x}_*), \sigma(\mathbf{x}_*)), \tag{7.7}$$

where

$$\mu(\mathbf{x}_*) = \mathbf{K_{*u}}\mathbf{A^{-1}}\mathbf{K_{uf}}\mathbf{Y}, \tag{7.8}$$

$$\sigma(\mathbf{x}_*) = \mathbf{K_{**}} - \mathbf{k_{*u}}(\mathbf{K_{uu}} - \mathbf{A^{-1}})\mathbf{K_{*u}^\mathrm{T}}, \tag{7.9}$$

where $\mathbf{K_{**}}$ is the kernel function evaluated at the test inputs, $\mathbf{K_{*u}}$ is the kernel function evaluated at the test and pseudo-point inputs.

7.5 Model Architecture

For the representation of goal-dependent hand movements, we devised a hierarchical model that consists of GP-LVMs for successive dimension reduction, and of a GPDM in the highest layer for the representation of the dynamics in the latent space (cf. Fig. 7.1a). The model is composed of three layers, which were learned jointly. Learning includes both the bottom-up and top-down contributions at each hierarchy level.

The proposed model can be run in two different modes: (i) As a *fully generative model*, where we the complete hand-arm kinematics is generated online, dependent on an external variable that controls the goal position; (ii) as a *partially generative model*, where the goal position is inferred online from measured arm kinematics and/or EMG signals. In this case these variables are treated as observed nodes in the graphical model, and the other variables are inferred by Bayesian model inversion. For the separation of style and content [113], style referring here to the goal position, we introduced a specific goal/style variable $\mathbf{g}$ that captures specifically the goal position, which either is pre-specified in the fully generative mode, or inferred in the partially generative mode.

7.5.1 Individual layers

Bottom layer.

The bottom layer of our model represents the observed kinematic data of the hand, the kinematic velocity data of the arm, and the EMG data as matrices $\mathbf{Y}^{(i)}, i \in \{1;2;3\}$. The dimensionality of each of these data sets is reduced by a sparse GP-LVM with the same sample size ($N^{(i)} = 1440$), but different dimensionalities of the data vectors ($D_{\mathbf{y}}^{(1)} = 63, D_{\mathbf{y}}^{(2)} = 15, D_{\mathbf{y}}^{(3)} = 28$). For each data type, a separate set of mapping functions is used with their own GP prior $f^{(i)}(\cdot)$ to map the latent variables $\mathbf{h}^{(i)}$ onto the corresponding data vectors, according to equation (3.13). The dimensionalities of the hidden variable vectors are $Q_{\mathbf{h}}^{(1)} = 10$, $Q_{\mathbf{h}}^{(2)} = 4$, and $Q_{\mathbf{h}}^{(3)} = 15$. The kernel functions in this layer are given by a linear combination of the RBF and the linear kernel functions defined in equations (3.9),(3.10) to capture the nonlinearity of the data and support smooth blending between styles:

$$k_{\mathbf{f}}^{(i)}(\mathbf{h}^{(i)}, \mathbf{h}'^{(i)}) = k_{rbf}(\mathbf{h}^{(i)}, \mathbf{h}'^{(i)}) + k_{lin}(\mathbf{h}^{(i)}, \mathbf{h}'^{(i)}). \tag{7.10}$$

Intermediate layer.

For further dimensionality reduction, an additional sparse GP-LVM was introduced that maps from the latent variable $\mathbf{x}$ ($Q_{\mathbf{x}} = 2$) onto the concatenation of the variables $\mathbf{h}^{(i)}$. We assume that $\mathbf{h}^{(i)}$ is represented in the same latent space across trials, number of goals, and time steps.

A key assumption in our approach is that the goal positions can be modeled by a separate 'style variable' $\mathbf{g}$. During the learning of the model parameters, we separate the motion content, i.e. the basic shape of the trajectories, from this motion style variable, using a factorial representation. For the partially generative model, we infer the style variable $\mathbf{g}$ online from the measured EMG and kinematic data of the arm. To promote such a factorization of the latent variables in terms of motion content and style dimensions, we applied *back-constraints* during learning [66]. This method constrains the hidden space variable to a low-dimensional manifold in a way that ensures that close points in the data space remain close in the latent space. Adjusting the gradient steps during optimization, we enforced the components of $\mathbf{x}$ to lie on a circular manifold for all styles that are given by the functions z_1 and z_2 [108]:

$$x_{n,1} = z_1([\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}]_n, \phi_n) = \sum_{\tau=1}^{N} c_{\tau,1} k_{rbf}(\cos(\phi_n), \cos(\phi_\tau)) \tag{7.11}$$

$$x_{n,2} = z_2([\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}]_n, \phi_n) = \sum_{\tau=1}^{N} c_{\tau,2} k_{rbf}(\sin(\phi_n), \sin(\phi_\tau)) \tag{7.12}$$

The variable ϕ_n specifies the motion phase, and the coefficients $c_{\tau,i}$ were chosen to ensure that the latent points lie (approximately) on a circle for all motion 'styles'.

The motion style variable $\mathbf{g}$, which specifies the reaching goal, was initially specified using 1-of-K encodings [6], each goal being specified by a (eight-dimensional) unit vector (goal 1: $\mathbf{g}_1$, goal 2: $\mathbf{g}_2$, etc.) which will also be optimized. In order to blend linearly between the behaviors for the different goals, we used a linear kernel for the 'style' dimensions. The composite kernel function for the middle layer of our model was thus given by (equations (3.9),(3.10)):

$$k_{\mathbf{h}}([\mathbf{x},\mathbf{g}],[\mathbf{x}',\mathbf{g}']) = k_{lin}(\mathbf{g},\mathbf{g}')k_{rbf}(\mathbf{x},\mathbf{x}') + k_{lin}(\mathbf{x},\mathbf{x}'). \tag{7.13}$$

Similar to the multi-factor model [113], the multiplication of style and latent position act similarly to a logical 'AND' operation. The resulting kernel matrix links each goal point to a pair of latent points.

Top layer.

The top layer represents the temporal evolution of $\mathbf{x}_n$, i.e. the underlying dynamics. We used a second-order GPDM, which allowed us to model the dynamical dependencies on velocity and acceleration of $\mathbf{X}$. The evolution function for $\mathbf{x}_n$ is given by equation (3.19). We drew the function $f_{\mathbf{x}}(\cdot)$ from a GP with the kernel given in equation (3.21), where we learned the model with multiple motion sequences (T trials for G goals, see Fig. 7.1a), assuming they are i.i.d. samples from the complete model.

In order to accelerate the inference of the partially generative model, we introduced an explicit learning of the back projections from the hidden variables $\mathbf{h}^{(2)}$ and $\mathbf{h}^{(3)}$ that represent the EMG and the arm kinematics to the variable $\mathbf{h}^{(1)}$ that represents the hand kinematics. These mappings were again represented by sparsely approximated Gaussian process regressions with the function priors $r_{\mathbf{y}}^{(j)}$, $j \in \{2,3\}$ (blue elements in Fig. 7.1b):

$$\mathbf{h}^{(j)} = r_{\mathbf{y}}^{(j)}(\mathbf{y}^{(j)}) + \boldsymbol{\epsilon}^{(j)}, \quad r_{\mathbf{y}}^{(j)}(\mathbf{y}^{(j)}) \sim GP(\mathbf{0}, k_{back}^{(j)}(\mathbf{y}^{(j)},\mathbf{y}'^{(i)})), \tag{7.14}$$

The kernel was given by $k_{back}^{(j)}(\mathbf{y}^{(j)},\mathbf{y}'^{(i)}) = k_{rbf}^{(j)}(\mathbf{y}^{(j)},\mathbf{y}'^{(i)}) + k_{lin}^{(j)}(\mathbf{y}^{(j)},\mathbf{y}'^{(i)})$, and $\boldsymbol{\epsilon}^{(j)}$ is Gaussian noise. As for a usual GP regression, only the kernel parameters are optimized during this step, while the latent variables for both layers were kept fixed. In an equivalent way, we introduced explicit back-projections for the second layer to predict the style variable $\mathbf{g}$.

7.5.2 Motion generation and prediction.

In the fully generative mode, our model, after training, predicts the hand kinematics (and also the arm kinematics and EMG signals) from a defined goal position. In the partially generative mode, it generates the hand kinematics from the arm kinematics and/or the EMG signals, estimating the likely goal position $\mathbf{g}$ from the available input data, while exploiting the back-projections for fast inference.

In both cases, the top dynamic layer ensures the generation of smooth hand motion. Given the states for the previous time steps, the GPDM predicts a novel state according to equation (3.19):

$$\mathbf{x}_n \cong \mu_{\mathbf{x}}([\mathbf{x}_{n-1},\mathbf{x}_{n-2}]). \tag{7.15}$$

In the fully generative mode, the predicted state is propagated down the hierarchy to predict the original data components $\tilde{\mathbf{y}}_n^{(i)}$ according to the relationships:

$$\mathbf{h}_n^{(i)} \cong \mu_{\mathbf{h}}([\tilde{\mathbf{x}}_n,\mathbf{g}_*]), \tag{7.16}$$

$$\mathbf{y}_n^{(i)} \cong \mu_{\mathbf{y}}(\tilde{\mathbf{h}}_n^{(i)}). \tag{7.17}$$

The functions $\mu_i(.)$ are following from the posteriors of the individual layers. The goal information is added through the variable $\mathbf{g}_*$ in this equation, for which values can be chosen that interpolate between the training goals' vectors, resulting in continuously interpolated intermediate goal positions.

In the partially generative mode, the goal vector $\mathbf{g}$ is predicted from the measurements $\mathbf{y}_*^{(j)}$, $j \in \{2, 3\}$, through the back-projections, according to the relationships:

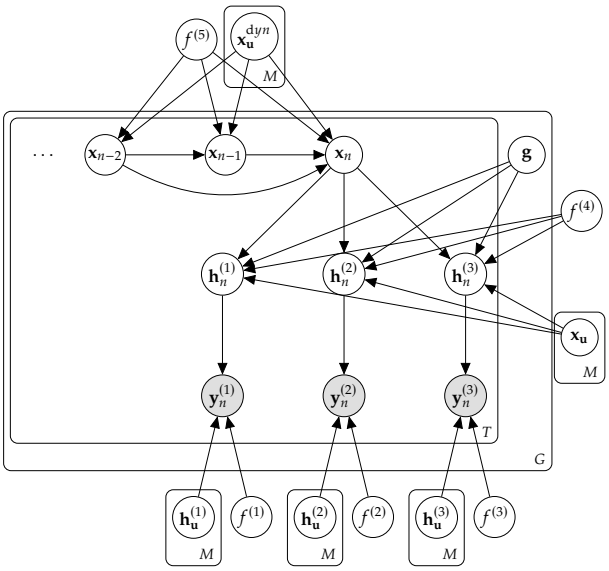
$$\mathbf{h}_n^{(j)} \cong \mu_{\mathbf{h}}(\mathbf{y}_*^{(j)}) \tag{7.18}$$

$$\mathbf{g} \cong \mu_{\mathbf{g}}^{(j)}(\tilde{\mathbf{h}}_n^{(j)}) \tag{7.19}$$

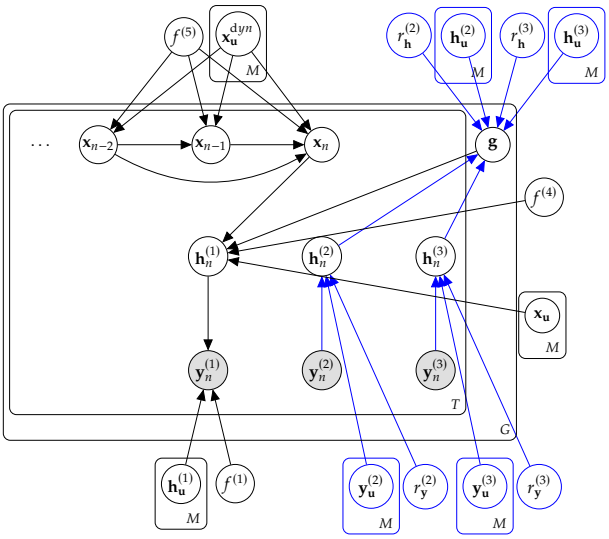
For the models with two predictive variables $\mathbf{h}_n^{(j)}$, the estimates were combined using a maximum a posteriori framework. The style vector estimate is then combined with the predicted state $\tilde{\mathbf{x}}_n$ from the dynamics and projected down to predict the hand kinematics $\tilde{\mathbf{y}}_n^{(1)}$ according to:

$$\mathbf{h}_n^{(1)} \cong \mu_{\mathbf{h}}([\tilde{\mathbf{x}}_n, \tilde{\mathbf{g}}]), \tag{7.20}$$

$$\mathbf{y}_n^{(1)} \cong \mu_{\mathbf{y}}(\tilde{\mathbf{h}}_n^{(1)}). \tag{7.21}$$



(a) Graphical model of the hierarchical GPDM.



(b) Back-projection of the hierarchical GPDM.

Figure 7.1: *Hierarchical probabilistic model.* (*a*) Three data structures $\mathbf{Y}^{(i)}$, $i \in \{1,2,3\}$ including the arm kinematic data, the hand kinematic data, and the EMG data are represented by sparse GP-LVMs, each over T trials and G goal positions. Each data structure is mapped by a prior mapping function $f^{(i)}(\cdot)$ drawn from a GP prior with a lower-dimensional latent variable $\mathbf{h}^{(i)}$. For each a set M pseudo-input variables $\mathbf{h}_{\mathbf{u}}^{(i)}$ are specified for sparse approximation. The dimensionality of the variables $\mathbf{h}^{(i)}$ on the middle level of the model is further reduced by a sparse GP-LVM in the same way, with the hidden state variables $\mathbf{x}$, the corresponding inducing variables $\mathbf{x}_{\mathbf{u}}^{(i)}$, and the style variable $\mathbf{g}$. The temporal evolution of the hidden state variables $\mathbf{x}_n$ is modeled with second order dynamics using a GPDM also drawn from a GP prior $f^{(5)}$. Corresponding inducing variables for the sparse representation are signified by $\mathbf{x}_{\mathbf{u}}^{dyn}$. (*b*) Back-projection mapping (blue) from data space of $\mathbf{y}^{(j)}$, $j \in \{2,3\}$ to the latent spaces $\mathbf{h}^{(j)}$, using sparse GP regressions with prior functions $r_{\mathbf{y}}^{(j)}$ and the pseudo-inputs $\mathbf{y}_{\mathbf{u}}^{(j)}$. Similarly, a back-projection from $\mathbf{h}^{(j)}$ to $\mathbf{g}$ is computed with the funcion prior $r_{\mathbf{h}}^{(j)}$ and pseudo-input $\mathbf{h}_{\mathbf{u}}^{(j)}$.

7.6 Results

7.6.1 Data set.

In order to test the performance of our approach, we applied it to a data set with grasping movements that included kinematic data from the arm and the hand, and HD-sEMG data from 3 high-density 8 x 8 surface electrode arrays (OT Bioelettronica) placed around the forearm, and from 16 bipolar sEMG electrodes on the larger muscles of the upper arm, shoulder, chest, and back. EMG recordings were done with an OT Bioelettronica Quattrocento 400 channel desktop amplifier (OTBioelettronica, Torino, IT) and three EMG pre-amplifiers. Samples were taken at 2,848 Hz, with an amplification gain of 150 V/V, and an online bandpass filter between 10 to 900 Hz. MOCAP recordings were conducted through VICON Nexus version 2.8 using eight VICON infrared cameras at 256.41 Hz.

Reaching movements were performed by a healthy, left-handed, adult male (39 years old, 178 cm tall, 82 kg) with no prior record of neuromuscular disease. Grasped objects were mounted on a vertical panel in the form of a rectangular 3×3 grid with a distance of 25 cm between the objects. All movements started from a resting position next to the hip.

EMG signals were processed by digital band-pass filtering between 20-450 Hz with a fourth-order Butterworth filter. A second-order Butterworth filter was applied as a band-stop filter between 49-51 Hz to remove power line interference. The signals were full-wave rectified and processed with a low-pass, second-order Butterworth filter at 10 Hz to extract the linear envelope.

7.6.2 Model evaluation.

We applied our method to infer the hand kinematics (63 DOFs) from either the arm kinematics (velocities) (15 DOF), the EMG signals, or from both (see Fig. 7.2). In total, we learned 16 motion trajectories. Due to outliers in the EMG data, we selected only two trials per goal for learning and one trial per goal for testing. Because of outliers in each trial we excluded goal two completely. Our raw training set consisted of EMG data, and arm and hand kinematics for eight goals.

The bvh kinematic arm and hand data were converted from radian to exponential map [39]. Then, we extracted the velocities from exponential map representations of the arm kinematics. We trained the model with 200 pseudo-inputs for each latent space on an AMD Ryzon Threadripper 1950X 16 core processor 3.4 GHz with 64 GB RAM, which took about 10.5 hours for 1500 iteration steps.

We computed the normalized Mean Square Prediction Error (NMSPR) for predictions of the hand kinematics (joint angles). Even for this small data set, the model produced acceptable hand postures during reconstruction and the NMSPRs for the hand kinematics were smaller for predictions from the arm kinematics than for

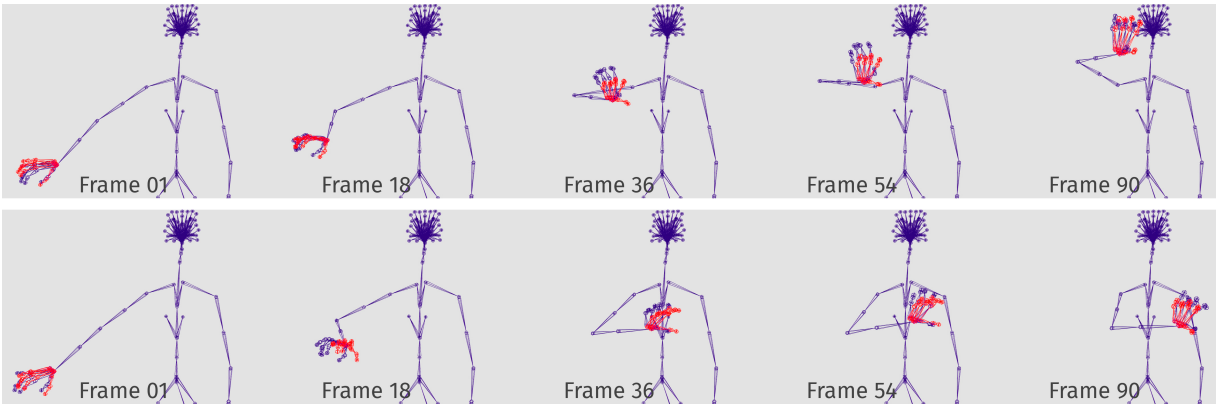


Figure 7.2: *Frames from animations illustrating the results.* Prediction examples of hand and finger orientations over time for goal 1 (top row) and goal 09 (bottom row). Predicted hand kinematics are shown in red, and the corresponding ground truth data are in blue.

Table 7.1: *Normalized Mean Square Prediction Error (NMSPR) for hand kinematics and goal position.* Prediction from arm velocities results in the smallest prediction error.

Hand NMSPR			
Goals	EMG + Velocity	EMG	Velocity
Goal 01	0.2425	0.2549	0.2472
Goal 03	0.2380	0.2852	0.2477
Goal 04	0.3692	0.4450	0.2795
Goal 05	0.3368	0.3851	0.3109
Goal 06	0.2865	0.3657	0.2436
Goal 07	0.3701	0.4206	0.3191
Goal 08	0.3920	0.4153	0.2992
Goal 09	0.2120	0.2091	0.1802
Average	0.3059	0.3476	**0.2659**

Goal NMSPR			
Goals	EMG + Velocity	EMG	Velocity
Goal 01	0.3317	0.2885	0.3377
Goal 03	0.2964	0.3150	0.2578
Goal 04	0.3682	0.4302	0.2860
Goal 05	0.2488	0.4081	0.3458
Goal 06	0.3463	0.4081	0.2631
Goal 07	0.3508	0.4874	0.3563
Goal 08	0.4040	0.4813	0.3170
Goal 09	0.2864	0.2561	0.1918
Average	0.3291	0.3844	**0.2944**

predictions from the EMG. Combining the prediction from EMG and arm kinematics reduced the prediction error substantially compared to the prediction from EMG alone.

We also analyzed the classification of the goal from the last 20 stimulus frames based on the latent space variable $\mathbf{g}$. The predicted classification was determined by the largest element of the inferred variable, which corresponds to the most likely class. The right part of Table 7.1 shows the NMSPR for the resulting predicted vectorial binary classification vectors relative to the ground truth, which is given by the true classes defined by the individual goals. Also for this measure, classification is best from the arm kinematics, and including the arm kinematics in the prediction improves accuracy compared to predictions derived from EMG alone.

An additional important point is that the proposed algorithm achieves *real-time performance* for online prediction of the hand kinematics. Running the algorithm on the same computer used for optimization, the prediction time per frame was on average about 23.74 ms.

7.7 Conclusion

Combining several methods from Bayesian unsupervised learning and inference, we have devised a new real-time-capable method for the simulation of reactive hand movements controlled by EMG. The high flexibility of the underlying inference framework was exploited to combine the EMG with additional kinematic data from the arm. Our model successfully predicted hand positions relative to the goal objects and reproduced hand kinematics with acceptable accuracy. Predictions derived from arm kinematics were more accurate for the tested data set than were the reconstructions from EMG. In addition, the accuracy of the reconstruction from EMG could be significantly improved by adding information from arm kinematics. The superiority of arm kinematics might be a consequence of our data set, which did not include transitions between different grip types, or supination movements. Such a variation in the dataset might give predictive power to the forearm HD-sEMG signals, enabling the disambiguation of movements that cannot be derived from the arm kinematics alone.

Despite the sophistication of the underlying probabilistic model, we have demonstrated that the algorithm is real-time-capable on a standard hardware. To our knowledge, underlying, advanced statistical methods such as GPDMs or inference on hierarchical GP-LVMs have never been tested and prepared for real-time applications. In our implementations real-time capability could be achieved by inclusion of several additional approximations, such as explicit back-projections and sparse approximations.

Future work will extend such architectures by applying variational optimization with better approximation methods to enable deep architectures with increased scalability. This will enable testing of these algorithms on much bigger data sets. Furthermore, we plan to implement the architecture with GPU computing for faster learning and inference.

Acknowledgments

We would like to thank Albert Mukovskiy for his invaluable insight and code regarding the pre-processing of our EMG and kinematic data. This research was funded through HFSP RGP0036/2016, BMBF FKZ 01GQ1704, KONSENS-NHE BW Stiftung NEU007/1, DFG GZ: KA 1258/15-1, ERC 2019-SyG-RELEVANCE-856495.

8. Interlude 2: Developing the GP BC

Although hierarchical GPDMs proved effective for prosthetic control in reach-to-grasp tasks, we found that GPDMs struggle to accommodate multiple distinct classes or categories of movements together. When trajectories from different movement classes overlap in the latent space, the resulting predictions become unreliable, causing unintended switching or morphing between movements. This challenge led us to develop a GPDM model capable of distinguishing and generating multiple movement types. In order to do this, I first needed a framework that would enable me to efficiently test a variety of architectures.

I chose to build this framework on top of GPy [38], a popular open-source Gaussian process Python package which includes a comprehensive set of Gaussian process implementations like GP regression, GPLVMs, back-constraints, sparse approximation techniques, a diverse set of kernels, and a range of optimization algorithms. My package extension significantly expanded GPy's capabilities, incorporating: hierarchical GPLVMs, node-based computation (which enables parallel processing in some hierarchical architectures), a novel, modular back-constraint framework, GPDMs, GPDM sparsification, GPDM mixture models, and comprehensive initialization procedures–all formatted to enable flexible architecture implementation. Additionally, we integrated into this framework all of the tangential components necessary for evaluating the performance of dynamical models. This includes a data set class code base that manages pre-processing, conversions to different data formats, calculating performance metrics, plotting latent spaces, animating 2D and 3D articulated human motion sequences, and data storage. Our GPy extension also includes a sub-package for statistical tests, including the parallel processing of simulations, Bayesian and grid search hyperparameter optimization, and Monte Carlo cross-validation. This considerable development effort provided the necessary tools for the systematic exploration of different model architectures and optimization strategies.

With my framework in place, I developed the first version of the GPDM mixture model, later referred to as the Gaussian process dynamical mixture model (GPDMM) in *Single-Example Learning in a Mixture of GPDMs with Latent Geometries* [95], presented in section 11. Initial experiments revealed several critical insights. While the mixture approach successfully addressed the fundamental problem of movement class separation, the model's performance proved highly sensitive to latent space organization. Through systematic testing of different architectures, we found that single-layer mixture models with carefully designed initialization procedures and constraints consistently outperformed more complex hierarchical variants (a summary of these results is presented in section 11). This finding led us to focus on optimizing latent space structure rather than adding architectural complexity.

In particular, we found model performance was positively influenced by the inclusion of latent geometries–features made by constraining the latent space to a particular geometric conformation. For an intuitive example, imagine mapping latent points to fit on the curve of an ellipse, where each consecutive point along the curve represents the next time point in a sequence. Such a representation is particularly effective for cyclic movements, like gait, where the latent space should be a closed loop. For acyclic movements, similar geometric constraints remain effective as they encode fundamental movement priors–specifically, the temporal continuity and smoothness inherent in natural motion, where each pose evolves gradually from its predecessor. Despite the benefits, maintaining these geometric arrangements during model optimization presented its own challenge, leading to the development of the Gaussian process back-constraint (GP BC).

The GP BC emerged as a solution that bridges two critical findings: the importance of initial conditions and the value of geometric latent space features. Unlike traditional back-constraints, the GP BC constrains the latent space to carefully designed initial conditions while parameterizing geometric features, all to support our prior assumptions about movement data. This approach proved particularly valuable in scenarios with limited training data, where constraints act as regularizers that help prevent overfitting.

We tested a variety of model components in order to determine an optimal model architecture for single-example learning, including first versus second-order dynamics, multi-layer hierarchies, various BCs, and

initialization strategies. The results consistently favored a single-layer, mixture model with first-order dynamics and GP BCs.

Our finding that latent space initial conditions played a critical role in model performance led us to explore methods for optimizing latent space organization prior to model fitting. This investigation culminated in our work on information bottleneck-based [105] feature selection, which provides a principled approach to identifying and organizing the most informative latent space features. By applying information theoretic principles to latent space optimization, this work addresses the fundamental challenge of determining optimal initial conditions for GPDMs, complementing the GP BC's ability to maintain beneficial latent space organization during model training.

During this period of development, we recorded the bimanual (BM) data set (see section 4.2) that proved particularly valuable for testing our information bottleneck method. Unlike the CMU data set (see section 4.3), which contains many movements with distinct poses that can sometimes be classified based on a single frame, the BM data set deliberately features movements that all diverge from a common starting point. This specific characteristic makes classification more challenging and requires careful organization of the latent space to achieve good performance.

We present this work in *Variable Selection in GPDMs Using the Information Bottleneck Method* [94]. The next section (9) reproduces this work nearly verbatim from the original publication, with edits only to reduce redundancy within the broader thesis and to expand the discussion of the GP BC. The core of the work was done by me, with supervision by Prof. Dr. Martin Giese, and data collection again led by Dr. Leonardo Gizzi.

Variable Selection in GPDMs Using the Information Bottleneck Method

9.

9.1 Abstract

Accurate real-time models of human motion are important for applications in areas such as cognitive science and robotics. Neural networks are often the favored choice, yet their generalization properties are limited, particularly on small data sets. This paper utilizes the Gaussian process dynamical model (GPDM) as an alternative. Despite their successes in various motion tasks, GPDMs face challenges like high computational complexity and the need for many hyperparameters. This work addresses these issues by integrating the information bottleneck (IB) framework with GPDMs. The IB approach aims to optimally balance data fit and generalization through measures of mutual information. Our technique uses IB variable selection as a component of GPLVM back-constraints to reduce parameter count and to select features for latent space optimization, resulting in improved model accuracy.

9.2 Introduction

A high computational cost and an excess of hyperparameters limit the utility of the GPDM. To compensate, we integrate GPDMs and the information bottleneck (IB) framework to enforce optimal dynamical latent space development. The IB balances data fit with generalization using mutual information for cost and regularization [105]. Our objective is to simplify optimization by tightly constraining the set of learnable parameters and decreasing the number of necessary hyperparameters. We present here a novel technique that integrates IB variable selection with GPLVM back-constraints to enhance the model's ability to produce task-suitable latent spaces.

9.3 Background

9.3.1 GPLVMs, GPDMs, and BCs

For a general overview on GPLVMs, GPDMs, and BCs, see Background & Methods sections 3.2, 3.3, and 3.6.

9.3.2 Information Bottleneck Framework

The IB framework models a probabilistic mapping between the variable X and Y through an intermediate bottleneck variable Z. It is expressed as the minimization of $I(Z,X) - \beta \cdot I(Y,Z)$, where the mutual information $I(X,Z)$ measures the interdependence between random variables X and Z. Essentially, the inclusion of $I(Z,Y)$ encourages Z to predict Y, while the $I(X,Z)$ enforces a minimal knowledge of X, lowering redundancy. The hyperparameter $\beta > 0$ moderates the trade-off between the compression and fit [105].

9.4 Methods

9.4.1 Gaussian Process Back-Constraints

The GPDM can converge sub-optimally when overfitting the emission mapping relative to the state transitions. Geometric constraints Urtasun et al. [108] and smooth latent space initial conditions can correct this imbalance. BCs that simply maintain the latent space close to good initial conditions, like a kernel PCA, can dramatically improve performance for this reason.

The GP BC is a method that ensures the convergence of the latent space close to its initial conditions. The model's latent variables are first initialized with an arbitrary dimension reduction of the training data. A radial basis function (RBF) GP mapping from the data to the latent space is then optimized. We use the regressional posterior of the GP to provide our constraint during learning,

$$\mathbf{X} = \mathbf{K}_{Y,Y} \cdot \mathbf{K}^{-1} \cdot \mathbf{X_0}, \tag{9.1}$$

where $\mathbf{X}$ is a matrix of latent variables, $\mathbf{K}^{-1}$ is the inverse learned GP kernel matrix, $\mathbf{K}_{Y,Y}$ is the kernel mapping of the training data with itself, and $\mathbf{X_0}$ is the initialized latent matrix. Optimization of the latent space is thus performed through learning the kernel parameters in $\mathbf{K}_{Y,Y}$ (and not in $\mathbf{K}^{-1}$).

After optimization, predictive mapping into the latent space is performed by equation equation 9.1 updated as,

$$\mathbf{X}^* = \mathbf{K}_{Y,Y^*} \cdot \mathbf{K}^{-1} \cdot \mathbf{X_0}, \tag{9.2}$$

where $\mathbf{Y}^*$ designates a set of testing data.

9.4.2 Latent Space Embedded Geometric Features

The following section is not presented in the original paper [94] due to space constraints, but is included here to elaborate on the GP BC approach .

Previous work used back-constraints (BCs) to enforce specific topologies in latent spaces for various applications. Urtasun et al. [108] proposed this approach for representing periodic data on a unit circle, while we applied it to non-periodic motions in a hierarchical model in section 7 [104]. Although this form of BC works well for a hierarchical model, it is not suitable for multi-class, single-layer models as it compresses the latent space into a two-dimensional elliptical conformation, good for dynamics but not for classification. The GP BC was designed to do both. In addition to the benefits described in the previous section 9.4.1, the GP BC incorporates geometric features using a novel method, defining parameterized geometries in $\mathbf{X_0}$ that are optimized alongside the kernel parameters in $\mathbf{K}_{Y,Y}$. The following two parameteric equations for an ellipse and a torus represent successful applications of this method: For an ellipse parameterized by semi-major axis R and semi-minor axis r, the parametric form is:

$$\mathbf{x}(\boldsymbol{\theta}) = R\cos(n\boldsymbol{\theta}), \quad \mathbf{y}(\boldsymbol{\theta}) = r\sin(n\boldsymbol{\theta}) \tag{9.3}$$

where $\boldsymbol{\theta}$ is the parametric angle and n is a parameter that determines the frequency of turns around the ellipse. The parametric equations for a torus are:

$$\begin{aligned} \mathbf{x}(\boldsymbol{\theta}) &= (R + r\cos n\boldsymbol{\theta})\cos\boldsymbol{\phi}, \\ \mathbf{y}(\boldsymbol{\theta}) &= (R + r\cos n\boldsymbol{\theta})\sin\boldsymbol{\phi}, \\ \mathbf{z}(\boldsymbol{\theta}) &= r\sin n\boldsymbol{\theta}, \end{aligned} \tag{9.4}$$

where R defines the outer radius of the torus, r the inner radius, and n represents the number of loops around the inner radius. We found that constraining $\boldsymbol{\theta} = \boldsymbol{\phi}$ led to a better representation of the dynamics.

We mapped our data trajectories to the variable $\boldsymbol{\theta}$ via the same method as presented in the following chapter under section 11.4.2. I have adapted it below for easy reference:

We construct a one-dimensional progression vector $\boldsymbol{\theta}$ for each sequence, running from 0 to 2π; the step size between consecutive data points is inversely proportional to the velocity of trajectory at this point. Thus, points with higher velocities in the original data receive smaller $\boldsymbol{\theta}$ increments, improving coverage in regions sampled sparsely over time. By applying eqs. 9.3, 9.4, or another parametric equation to these cumulative $\boldsymbol{\theta}$ values, we obtain a set of d-dimensional (where $d = 2$ and $d = 3$ for the ellipse and torus, respectively) vectors that form the matrix $\mathbf{X_G}$.

Finally, we combine $\mathbf{X_G}$ with features from a suitable dimensionality reduction of the data (e.g., PCA), denoted $\mathbf{X_R}$, to initialize the latent variables: $\mathbf{X_0} = \left[\mathbf{X_G}, \mathbf{X_R}\right] \in \mathbb{R}^{N\times Q}$ where N is the total number of data points and Q is the combined dimension of $\mathbf{X_G}$ and $\mathbf{X_R}$. This enables us to represent the data's dynamics via a learned, smooth geometry while leveraging low-dimensional embeddings of the original dataset for improved classification.

9.4.3 Information Bottleneck Latent Optimization

IB latent optimization (IBLO) begins by defining a latent space with more than the expected number of task-optimal features. Here, we designate $\mathbf{Y}$ as the data space and $\mathbf{X}$ as the high-dimensional latent space. We aim to optimize an auto-regressive mapping from $\mathbf{y}_t$ to $\mathbf{y}_{t+1}$.

Within this framework, we optimize a GP RBF mapping from $\mathbf{X}$ to $\mathbf{Y}$. We then establish an auto-regressive mapping in $\mathbf{X}$ using the same type of kernel as in our GPDM dynamics. These GPs furnish us with the probability distributions: P($\mathbf{X}$) and P($\mathbf{Y}|\mathbf{X}$). The former characterizes the auto-regressive dynamics of $\mathbf{X}$, while the latter provides a GP regression from $\mathbf{X}$ to $\mathbf{Y}$.

Our central objective is to define $\mathbf{Z}$ as the random variable to be optimized, serving as the bottleneck between $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$. We aim to identify a subset, denoted as $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots \mathbf{x}_d \subseteq \mathbf{X}$, for $\mathbf{Z}$ that includes only the dimensions essential for describing $\mathbf{Y}$. To assess the relevance of dimension $\mathbf{x}_j$, we employ the following indicator function, inspired by the work in Pan et al. [78]:

$$S_j = [I(\mathbf{Y}_t, \mathbf{Z}_{t,+j}) - \beta I(\mathbf{Z}_{t+1,+j}, \mathbf{Z}_{t,+j})] - [I(\mathbf{Y}_t, \mathbf{Z}_{t,-j}) - \beta I(\mathbf{Z}_{t+1,-j}, \mathbf{Z}_{t,-j})] \tag{9.5}$$

Here, the subscripts $-j$ and $+j$ signify whether $\mathbf{z}_j$ is not in $\mathbf{Z}$ or is included in $\mathbf{Z}$, respectively. $I(\mathbf{N}, \mathbf{M})$ represents the mutual information between $\mathbf{N}$ and $\mathbf{M}$, calculated using the multivariate Gaussian entropy. S_j is a measure of how well $\mathbf{Z}$ serves as a bottleneck when either retaining or eliminating dimension $\mathbf{x}_j$. The greater the value of S_j, the more $\mathbf{Z}$ will benefit from the removal of $\mathbf{x}_j$. S_j is evaluated iteratively on successive removals of dimension j. We set β by solving for $S_j = 0$ on the evaluation of the first j in the first iteration. This enabled the algorithm to optimally balance features that were pertinent to both compression and dynamical prediction.

9.5 Results

We evaluated the IBLO method by its ability to accurately categorize actions under different conditions. This classification was accomplished by comparing the posterior probabilities of a given sequence for different GPDMs. The data set comprised infrared motion capture of five bimanual manipulation tasks encompassing 117 degrees of freedom.

Our initial trials examined the ability of our IB method to select optimal RBF kernel PCA (kPCA) features compared to the model's performance when utilizing features selected according to the highest eigenvalues. In these cases, our method produced no significant improvement as it nearly always selected the highest eigenvalue features.

As a second comparison, we ran simulations where IBLO selected 10 features from a mixture containing 10 RBF kPCA features and 30 features produced by random projection—a dimensionality reduction technique wherein new features are produced by a linear combination of the features (in our case, Y) with weights sampled from a Gaussian distribution. We compared this against a control containing an equal number of features produced only by random projection. From these results, we derived two confusion matrices and accuracy scores based on the number of sequences correctly classified. A t-test comparing these accuracy scores gave a t-statistic of -5.83 with a p-value of $6 \cdot 10^{-7}$. Figure 9.1 displays a differential confusion matrix constructed from the IBLO scores minus those of the control. We further tested whether the IBLO method could select features better than expected for an arbitrary dimensionality reduction method. We tested the highest 10 eigenvalue features from PCA and ICA against a selection by IBLO from a set of the first 30 features of both techniques. In this case, IBLO did not use the same features as the control. Figure 9.2 shows the comparison for both ICA and PCA. In both cases IBLO improved the accuracy, precision, recall, and F1 score of the model.

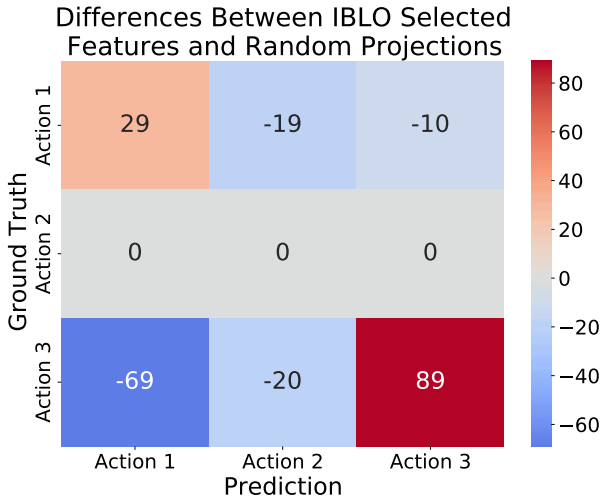


Figure 9.1: A heat map displaying the difference in sequence classification accuracy for three actions when comparing IBLO against the control. We constructed this heat map by subtracting the confusion matrix of the control from the confusion matrix for IBLO. Positive and negative values indicate areas where IBLO and the control, respectively, had greater scores. The numbers on the diagonal represent scores for the correct outcome, indicating IBLO had improved accuracy when classifying actions 1 and 3. For this test, IBLO selected 10 features from a mixture of 10 RBF kPCA and 30 random projections while the control used 10 random projections. Comparison of the two accuracy scores produced a significant difference with a p-value of $6 \cdot 10^{-7}$

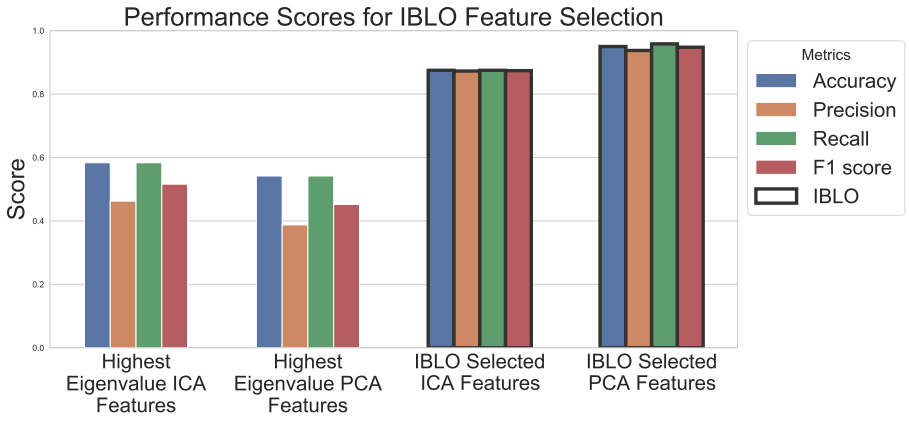


Figure 9.2: Performance scores for model classification of five sequences using PCA and ICA features. The control model used the first 10 features with the highest eigenvalues, while IBLO selected 10 from the first 30. In both PCA and ICA, IBLO improved performance on metrics of accuracy, precision, recall, and F1 score. Further simulations must be run to determine a significance score, but results suggest a continuation of the displayed trend.

9.6 Conclusion

The results are promising, showing an ability for IBLO to select highly informative kPCA features from sets intermixed with random projections. It also shows a trend towards improved performance when comparing PCA and ICA features chosen according to the highest eigenvalues to those selected by IBLO. Future research will continue to test the performance and utility of the model.

Interlude 3: Formal Introduction and Optimization of the GPDMM Architecture

10

Following our investigations into feature selection using the information bottleneck method, we conducted further analyses of GPDM mixture model architectures using an improved set of performance metrics that more comprehensively evaluated the model's generative capabilities. This systematic evaluation revealed several unexpected insights that challenged our initial assumptions about model architecture and led to significant improvements in performance.

Our most striking finding concerned the role of geometric features in latent space organization. While previous work had emphasized the importance of BCs for enforcing specific topologies, our experiments demonstrated that incorporating geometric features directly into the initial conditions of the latent space was not only sufficient but superior to enforced constraints. We present a thorough evaluation of this work in the next section (11) and the accompanying appendix (A).

This discovery led us to fundamentally rethink how geometric information should be incorporated into GPDMs. We found that instead of actively enforcing geometric constraints during optimization, simply providing appropriate geometric structure during initialization was sufficient to guide the model toward optimal latent space organizations. This approach's superior performance appears to stem from two key advantages. First, initializing with geometric features gives the optimization process a well-structured starting point while preserving the model's flexibility during learning. Second, removing explicit constraints reduces model complexity, thereby limiting the number of potential local optima.

Through this work, we further streamlined our GPDM mixture model by removing BCs while preserving first-order dynamics and a single-layer architecture. We named this final version the Gaussian process dynamical mixture model (GPDMM). This refined model served as the foundation for our comprehensive evaluation, which compared performance across different latent space initial conditions, architectural variations, and contemporary deep learning approaches. Our paper *Single-Example Learning in a Mixture of GPDMs with Latent Geometries* [95] is presented in the next section. Unlike the previous two papers that were abridged, I have preserved most of the content verbatim from the original publication to provide a thorough overview of the model and its variants, as this work represents the culmination of this thesis. Additionally, I include a detailed appendix A not in the original publication, which presents an expanded evaluation of the model's performance across latent space initial conditions. As with the previous chapter, I conducted the core of the work under the supervision of Prof. Dr. Martin Giese, with Dr. Leonardo Gizzi leading the BM experiment data collection.

Single-Example Learning in a Mixture of GPDMs with Latent Geometries

11.

11.1 Abstract

We present the Gaussian process dynamical mixture model (GPDMM) and show its utility in single-example learning of human motion data. The Gaussian process dynamical model (GPDM) is a form of the Gaussian process latent variable model (GPLVM), but optimized with a hidden Markov model dynamical prior. The GPDMM combines multiple GPDMs in a probabilistic mixture-of-experts framework, utilizing embedded geometric features to allow for diverse sequences to be encoded in a single latent space, enabling the categorization and generation of each sequence class. GPDMs and our mixture model are particularly advantageous in addressing the challenges of modeling human movement in scenarios where data is limited and model interpretability is vital, such as in patient-specific medical applications like prosthesis control. We score the GPDMM on classification accuracy and generative ability in single-example learning, showcase model variations, and benchmark it against LSTMs, VAEs, and transformers.

11.2 Introduction

Modeling human motion remains a significant challenge across neuroscience, medicine, computer graphics, and robotics due to data scarcity, high dimensionality, motor redundancy, and the requirement for real-time computation [117]. While neural networks are commonly employed, their effectiveness depends on large training sets that are often difficult or impossible to obtain, particularly in specialized applications. Gaussian processes (GPs), namely the Gaussian process dynamical model (GPDM), offer an effective alternative by enabling accurate human movement synthesis from limited data [111, 112].

The GPDM is an extension of the Gaussian process latent variable model (GPLVM) with a hidden Markov model (HMM) prior. Decomposed into two GPs, the emission GP, like in the standard GPLVM, maps the latent space to the observed data, while the dynamical GP models the temporal evolution of the latent space as an HMM. It is effective in low-dimensional representation, motion modeling, and sequence prediction [111, 112]. Additionally, sparse approximation techniques have made GPDMs both scalable and suitable for real-time applications [64, 93]. Since GPDMs are non-parametric, they are more robust to model misspecification than parametric models, and can better leverage limited data.

GPDMs also offer superior interpretability compared to black-box neural models. Its latent space provides a visualizable low-dimensional representation, its structure is easy to explain with a straightforward Markovian prior, and its probabilistic formulation explicitly quantifies uncertainty in both the dynamics and observations [111, 112]. These features are critical for applications requiring transparency such as medical diagnosis or prosthesis control.

Despite the many advantages of GPDMs, they have distinct weaknesses. They cannot explicitly classify movements (though they may do so implicitly by predicting the dynamics of a particular class correctly). Furthermore, our own experiments show that they struggle to robustly handle the prediction of multiple movement types simultaneously. This limitation stems from their small temporal prediction horizon, which causes ambiguity when movements intersect or converge in the latent space, leading to inappropriate morphing or switching between classes. Additionally, we found that even when modeling a single movement type, GPDMs yield unstable predictions over longer time horizons due to their step-by-step Markovian prediction process, which accumulates errors that cause trajectories to drift from their true paths.

To address the challenge of modeling diverse movements with limited training examples, we propose the Gaussian process dynamical mixture model (GPDMM), which applies the mixture-of-experts approach [49] to GPDMs. The GPDMM comprises a probabilistic mixture of dynamical GPs, where each GP acts as an expert on a particular movement class's dynamics, all unified by a single emission GP for positional representation. This architecture enables classification while preventing unintended switching or morphing between movements from these different classes. Central to the model's effectiveness is our method for embedding geometric features in the latent space, which enables smooth dynamics and stable long-horizon prediction. Significantly, we achieve high-quality performance using training data with only a single example per movement class, fulfilling one of our primary objectives.

In this paper, we present the formulation of the GPDMM and demonstrate its ability to classify and generate movements. We showcase the model's performance under design variations and ablations, and benchmark it against transformers, VAEs, and LSTMs.

11.3 Background

11.3.1 Gaussian Process Dynamical Models

The GPLVM [62] is a non-linear dimensionality reduction approach that maps each high-dimensional data point onto a lower-dimensional latent space via a GP prior. Building on the GPLVM, the GPDM [111, 112] adds a Markovian prior over the latent states, modeling temporal structure and enabling the generation of

new sequences. Furthermore, it allows us to evaluate how likely a new sequence is to have arisen from the trained distribution:

$$p\left(\mathbf{X}^* \mid a, \mathbf{X}_{in}, \mathbf{X}_{out}\right) \propto \exp\left(-\tfrac{1}{2}\operatorname{tr}\left[\mathbf{K}_{\mathbf{X}^*}^{(a)-1}\mathbf{Z}_{\mathbf{X}}^{(a)}\mathbf{Z}_{\mathbf{X}}^{(a)T}\right]\right), \tag{11.1}$$

where $\mathbf{X}^*$, $\mathbf{X}_{in}$ and $\mathbf{X}_{out}$ represent, respectively, the latent projection of the new data, and the latent input and output of the autoregressive dynamics. $\mathbf{K}_{\mathbf{X}^*}$ is a kernel matrix evaluated between $\mathbf{X}_{in}$ and $\mathbf{X}^*$. $\mathbf{Z}_{\mathbf{X}}$ is a mean function that includes $\mathbf{X}_{in}$, $\mathbf{X}_{out}$, and $\mathbf{X}^*$. Here, a is used as an indicator variable to specify a single GPDM in a mixture model. See Wang et al. [111, 112] for a full evaluation of the GPDM and eq. equation 11.1.

The computational cost of full GP inference scales cubically, $\mathcal{O}(N^3)$, where N is the number of training points. To mitigate this, sparse Gaussian process approximations introduce a smaller set of M inducing points ($M \ll N$) [93]. The fully independent training conditional (FITC) approach approximates the GP prior so that function values become conditionally independent given these inducing points, reducing computational cost to $\mathcal{O}(NM^2)$ while preserving much of the GP's flexibility [64]. Although the data sizes used in these experiments are small enough to employ the full GPDMM with little cost, we build the FITC approximated GPDMM to measure its level of underfitting compared to the full GPDMM and to guide future research on GPDMMs with larger datasets.

11.3.2 Mixture of Experts

Multiple GPDMs can be integrated in a single model using a mixture-of-experts formulation [49], wherein each dynamical GP is trained only on a subset of the data determined by its class, $S_a \subset S$ where $a \in \{1, 2, ..., A\}$, A is the number classes, $|S_a| = n_a$, $|S| = N$, and N is the total number of training data points. For our applications, $n_1 = n_2 = ... = n_A = \frac{N}{A}$. Compared to a singular GPDM, this reduces the computational complexity of the dynamical component from $\mathcal{O}(N^3)$ to $\mathcal{O}(\frac{N^3}{A^2})$.

Given a subset of consecutive data points in a sequence (e.g. the first few seconds of a movement), the model can assign likelihoods for each possible category. In the following, we abbreviate eq. equation 11.1 by $p(\mathbf{X}^*|a)$. We use Bayes theorem to evaluate the posterior probabilities,

$$p(a|\mathbf{X}^*) = \frac{p(a)p(\mathbf{X}^*|a)}{\sum_{\alpha \in \mathcal{A}} p(\alpha)p(\mathbf{X}^*|\alpha)}, \qquad p(a) = \frac{n_a}{N} \tag{11.2}$$

where $\mathcal{A} = \{1, 2, ..., A\}$. A prediction is made by evaluating the argument of the maximum,

$$\arg\max_{\alpha \in \mathcal{A}} p(\alpha|\mathbf{X}^*) \tag{11.3}$$

11.4 Methods

11.4.1 Model Architecture

The basic GPDMM, like GPDMs, employs a single-layer architecture. The observation space of the model represents the observed variables (here joint-angle time series) and is denoted by a single data structure $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^T \in \mathbb{R}^{N \times D}$ where D is the number of features and N is the number of data points over all sequences. Each column in $\mathbf{Y}$ is composed of individual sequences stacked end-to-end for the length of the full data set.

The core of the model is an emission GP that maps a low-dimensional latent space to a higher-dimensional observable data space. Dynamics in the latent space are modeled as state transitions by G dynamical GPs, one for each category, g, of actions. Both the emission and dynamical GPs can be sparsified for scaling to

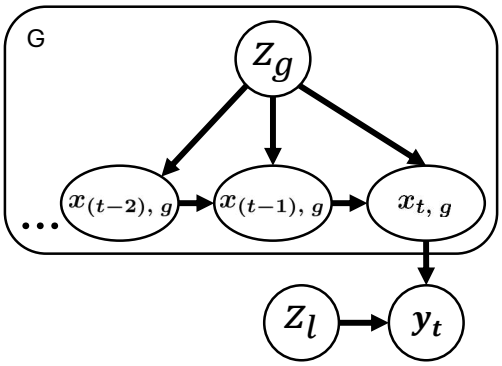


Figure 11.1: *GPDMM Graphical Model.* The data space represented by a single data structure $\mathbf{Y}$ comprises kinematics with N data points consisting of equal-length sequences with D feature dimensions. The data is represented by a sparse emission GP that maps from a lower-dimensional latent vector $\mathbf{x}_t$ at any point in t with inducing inputs $\mathbf{Z_l}$ representing the entire latent space. The latent dynamics are modeled as state transitions by G sparse dynamical GPs with inducing inputs $\mathbf{Z_g}$, each specializing in a distinct action category, g. Both $\mathbf{Z}$ variables are dropped for the non-sparse, full-inference model.

large data sets. Furthermore, the model can be expanded to multiple layers of GPLVMs, though it proved inefficient for our limited data sets (see table 11.2).

Implementation.

The GPDMM was built on top of the GPy library [38]. In particular, we used their core GPLVMs, kernel functions, and optimization procedures. The mixture model and the bulk of the dynamical GPs were produced in-house with some code heavily adapted from GPy, e.g., in applying sparse approximations. The full (non-sparse) model was about 7.5K parameters on the BM data set and 7.2K on the CMU data set.

Comparison to Switching GPDMs.

The GPDMM shares conceptual ground with Chen et al.'s switching GPDM [15]. However, it differs on several key points: first, the GPDMM emphasizes stable, long-horizon generation rather than short-term tracking. Second, it uses a fully probabilistic mixture-of-experts structure in a shared latent space, removing the need for discrete switching. Third, our model leverages direct likelihood evaluation for classification and generation, facilitating efficient inference. Crucially, the GPDMM supports single-example learning, embeds geometric features for smoother latent representations, and enables sparse approximations to scale effectively to larger datasets.

11.4.2 Initial Conditions and Latent Space Geometric Features

The optimization of GPLVMs is a highly non-convex problem, making solutions vulnerable to converging on local optima. Consequently, performance can depend significantly on the choice of the initial conditions for the optimization, particularly for the latent space [7, 67]. Latent space initialization is often done using a form of dimensionality reduction like PCA.

Previous work has employed back-constraints (BCs) to enforce specific topologies in latent spaces. For example, Urtasun et al. [108] used BCs to represent periodic data on a unit circle, and Taubert et al. [104] extended this idea to non-periodic motions in a hierarchical model. In our single-layer framework for dynamics, we compared both BC-based and unconstrained approaches, ultimately finding better performance with the unconstrained option (see table 11.2).

Fourier Basis as Latent Geometries.

The "unconstrained" approach embeds a geometry in the latent space via initialization with geometric features. Outlined below is our method for constructing these features, using the example of our top-performing geometry based on Fourier basis functions.

We defined a matrix of latent features $\mathbf{X_G}$ as a set of Fourier basis functions:

$$\mathbf{X_G} = \mathbf{f}(\boldsymbol{\theta}) = \left[\mathbf{1}, \quad \cos(2\pi\boldsymbol{\theta}), \quad \sin(2\pi\boldsymbol{\theta}), \quad \ldots, \quad \cos(m\pi\boldsymbol{\theta}), \quad \sin(m\pi\boldsymbol{\theta})\right], \tag{11.4}$$

where $\mathbf{1}$ is a vector of ones, and $\cos(\cdot)/\sin(\cdot)$ terms are included at frequencies from 1 to m. This setup yields a $(2m+1)$-dimensional representation. The hyperparameter m and the inclusion of the constant term were both optimized during model selection.

Mapping Data to the Latent Space.

We construct a one-dimensional progression vector $\boldsymbol{\theta}$ for each sequence, running from 0 to 2π; the step size between consecutive data points is inversely proportional to the velocity of trajectory at this point. Thus, points with higher velocities in the original data receive smaller $\boldsymbol{\theta}$ increments, improving coverage in regions sampled sparsely over time. By applying eq. 11.4 to these cumulative $\boldsymbol{\theta}$ values, we obtain a set of d-dimensional basis vectors that form the matrix $\mathbf{X_G}$.

Initialization of Latent Variables.

Finally, we combine $\mathbf{X_G}$ with features from a suitable dimensionality reduction of the data (see table 11.1), denoted $\mathbf{X_R}$, to initialize the latent variables: $\mathbf{X} = \left[\mathbf{X_G}, \mathbf{X_R}\right] \in \mathbb{R}^{N\times Q}$ where N is the total number of data points and Q is the combined dimension of $\mathbf{X_G}$ and $\mathbf{X_R}$. This initialization preserves the geometric relationships derived from the Fourier basis while leveraging low-dimensional embeddings of the original dataset.

Model Influence.

These geometric embeddings significantly boost performance (see table 11.1). We reason that this is explained by enforcing smoothness, capturing key periodicities, and allocating finer resolution to high-velocity segments, enabling coherent, comparable representations for diverse motion tempos within a shared latent space.

11.4.3 Benchmarking Model Architectures

The authors are unaware of a standard benchmark for the task of solving classification and long-horizon sequence prediction on small data sets of human movement. Given these limitations, we benchmarked the GPDMM against three popular sequence-generation and classification models: a VAE, Transformer, and LSTM network. Hyperparameter optimization, including variations of the model architectures, was performed consistently for all models (see section 11.5.2).

Variational Autoencoder (VAE).

Implemented in PyTorch [58, 79], the VAE features an encoder–decoder design with a reparameterizer (for mean and log variance) plus a classifier branch. It optimizes a combined loss of mean-squared error (reconstruction), KL divergence (latent space regularization), and cross-entropy (classification). This architecture was optimized over the number of hidden and latent dimensions. The number of parameters of the optimized VAE was roughly 12.3M and 7.6M, respectively, for the BM and CMU data sets.

Transformer.

Built via Hugging Face's BERT architecture [29, 116], the Transformer applies separate linear heads for classification (using the [CLS] token) and sequence generation. It uses cross-entropy for classification and mean-squared error for generation, optimized with Adam. This architecture was optimized over the number of hidden dimensions, layers, attention heads, and dropout rate. The number of parameters of the optimized Transformer was 19M and 15M, respectively, for the BM and CMU data sets.

Long Short-Term Memory (LSTM).

An LSTM layer (via PyTorch [45, 79]) feeds two linear heads: one for classification (final hidden state) and one for generation (all hidden states). Again, training combines cross-entropy and mean-squared error losses. This architecture was optimized over the number of layers and hidden dimensions. The number of parameters of the optimized LSTM was about 220K and 330K, respectively, for the BM and CMU data sets.

11.4.4 Resources

Code is available online at `https://github.com/jesse-st-amand/`.

All computations were performed on a single workstation with a AMD Ryzen 7 3700X 8-Core Processor and an NVIDIA GeForce RTX 2070 SUPER graphics card.

Claude 3.7 Sonnet [3] and ChatGPT o1 [76] were used in revising the text and for assistance in coding. The authors take full responsibility for the content of the manuscript.

11.5 Results

11.5.1 Data

We used two motion-capture data sets: (1) the *Carnegie Mellon University (CMU) data set* [21], comprising 6 trials each of 8 full-body movements (e.g., kicking, lunging, swimming), and (2) our *bimanual (BM) data set*, recorded in-house, featuring upper-body activities of daily living (e.g., lifting a box, rotating a tablet, opening a jar), performed while seated, and aligned to a universal starting position. Each sequence was converted to joint-angle representation at equal-length intervals. The CMU data contained 77 joint features (neck, spine, pelvis, arms, legs); the BM data contained 117 (upper body plus finger articulation).

11.5.2 Hyperparameter Optimization and Significance Testing

Bayesian hyperparameter optimization was performed individually per data set via the Python package Scikit-Optimize on all models used in our experiments. Within each iteration of the Bayesian search, we performed a limited Monte Carlo cross-validation (MCCV). At each iteration of the MCCV, we trained on a single example per class and validated and tested on the remaining sequences. For the BM set, this yielded 4 validation and 5 test examples per class; for the CMU set, 2 validation and 3 test examples per class. The validation sets were used to find the optimal number of training epochs before overfitting occurred. We averaged these optimal validation scores over MCCV iterations as input to our Bayesian model evaluation. Final model evaluation and significance testing was performed on the test sets.

Classification used only the first T elements of each test sequence, and generation was performed on the remaining elements. T was set to 40% and 15% of the sequence length for the BM and CMU data sets, respectively.

11.5.3 Scoring Procedure

To evaluate the GPDMM against other approaches, we computed metrics for both classification accuracy and sequence generation quality.

For scoring classification accuracy, we used the standard F1 score, calculating precision and recall across all classes.

To measure distances from the ground truth, we computed the Fréchet distance, which measures the distance between trajectories by minimizing the maximum point-wise distance between re-parameterizations of the curves, providing a natural control for time warping [32]. To account for differing classification accuracy, we only measured sequences that were correctly classified by that model.

We averaged and normalized our distance measures according to the following equation:

$$D_{avg} = \frac{1}{|C|} \sum_{c \in C} \frac{1}{|S_{c,v}|} \sum_{\{\mathbf{s}_{t,a}, \mathbf{s}_{t,b}\} \in S_{c,v}} D_N\left(g(\mathbf{s}_{t,a}), \mathbf{s}_{t,b}\right) \tag{11.5}$$

$$D_N(\mathbf{s}_g, \mathbf{s}_t) = \frac{d_F(\mathbf{s}_g, \mathbf{s}_t)}{\max_{\mathbf{s}_1, \mathbf{s}_2 \in \mathbf{S}_c} d_F(\mathbf{s}_1, \mathbf{s}_2)} \tag{11.6}$$

where C is the set of all classes, S_c is the set of all testing set sequences in class c, $S_{c,v}$ is the subset of those sequences correctly classified, $d_F(\mathbf{s}_1, \mathbf{s}_2)$ is the Fréchet distance between $\mathbf{s}_1$ and $\mathbf{s}_2$, $D_N(\cdot)$ is the normalized Fréchet distance, $\mathbf{s}_{t,a}$ is the ground truth sub-sequence used in classification, $\mathbf{s}_{t,b}$ is the remaining ground truth sub-sequence, and $\mathbf{s}_g$ is the sequence generated by $g(\mathbf{s}_{t,a}) = \mathbf{s}_g$ of size $\mathbf{s}_{t,b}$. This normalization controls for biases between sequence classes.

In addition to distance, we evaluated damping and smoothness metrics. Damping captures the reduction in motion amplitude and follow-through compared to natural movements. Smoothness quantifies the fluidity of a movement trajectory, with poor scores indicating jittery movements. These metrics address quality dimensions that F1 and distance metrics can miss. See table 11.2 row 7 (Fourier; None) for an example of a model that scored well on all metrics except for damping, and table 11.3 row 3 (VAE) for a similar example that scores poorly on LDJ.

For the damping metric, we analyze mean displacement over sliding windows calculated as,

$$d = \frac{1}{N - w} \sum_{i=1}^{N-w} \|\mathbf{p}_{i+w} - \mathbf{p}_i\| \tag{11.7}$$

where N is the number of points in the sequence, w is the window size, and $\mathbf{p}_i$ is the position of the i-th point. The damping metric is computed as a ratio between ground truth and generated sequences. Damping scores greater than 1 indicate that generated movements exhibit diminished amplitude or incomplete execution compared to the ground truth.

To quantify smoothness, we employed the log dimensionless jerk (LDJ) metric [4], which controls for duration and amplitude. The LDJ is defined as:

$$\eta_{LDJ} = -\ln\left(\frac{(t_2 - t_1)^3}{v_{\text{peak}}^2} \cdot \int_{t_1}^{t_2} \left(\frac{d^3x}{dt^3}\right)^2 dt\right) \tag{11.8}$$

where $t_2 - t_1$ represents the movement duration, v_{peak} is the peak velocity, and the integral represents the squared jerk over the movement interval. Higher (less negative) LDJ values indicate smoother movements. Similar to the damping metric, we calculate the LDJ ratio between generated and ground truth sequences, where values greater than 1 indicate degraded smoothness in the generated movements.

11.5.4 Experiments

Tables 11.1, 11.2, and 11.3 summarize model performances with *F1* indicating the F1 score, and *Distance*, *Damping*, and *LDJ* represented by eqs. equation 11.5, equation 11.7, and equation 11.8, respectively. An asterisk (*) denotes a statistically significant difference from the top performing model for that metric ($p < 0.05$). Values in bold indicate the best score per column. Values not significantly different from the top performing model are considered competitive. For both the damping and LDJ metrics, we consider scores closer to 1 to be better since this indicates a value closer to the ground truth. However, in-context we found LDJ scores below 1 to often be acceptable since this indicates trajectories with "less noise" than the ground truth–a possible indicator for over-smoothing that is (in)validated by the damping metric. Table 11.1

Table 11.1: GPDMM comparison for top-performing variations of geometries (geo) and dimensionally reduced features (DR) as initial conditions in the latent space. Dims refers to the number of latent dimensions occupied by the geometry where "NA" is not applicable and "all" is the total number of dimensions in the latent space. See chapter A, tables A.1 and A.2 for results on additional geos, DRs, and combinations not presented here.

		Dims		F1		Distance		Damping		LDJ	
Geo	**DR**	**BM**	**CMU**	**BM**	**CMU**	**BM**	**CMU**	**BM**	**CMU**	**BM**	**CMU**
None	Isomap	NA	NA	0.781*	0.996	0.233*	0.135*	3.551	1.398	1.026	0.902*
None	kPCA	NA	NA	**0.959**	**1.0**	0.232*	0.181*	6.44*	43.719	0.784*	0.754*
None	PCA	NA	NA	0.941	**1.0**	0.175*	0.148*	1.925*	3.551*	0.882*	0.785*
None	Random	NA	NA	0.829*	0.769*	0.359*	0.338*	94.513*	37.257*	**0.998**	**0.992**
None	UMAP	NA	NA	0.909	0.996	0.172*	0.138*	3.523*	1.716*	0.881*	0.772*
Chebyshev	None	All	All	0.943	**1.0**	0.145	0.126*	1.625*	1.782*	0.813*	0.816*
Fourier	None	All	All	0.938	**1.0**	**0.131**	0.11	1.624*	1.297*	0.807*	0.77*
Laguerre	None	All	All	0.911	**1.0**	0.152	0.136*	1.639*	1.915*	0.73*	0.784*
Ellipse	kPCA	2	2	0.955	**1.0**	0.151*	0.116	1.526*	1.084*	0.775*	0.718*
Fourier	kPCA	11	3	0.941	**1.0**	0.152*	0.115*	**1.104**	1.048*	0.943*	0.788*
Fourier	PCA	13	4	0.928*	**1.0**	0.174*	**0.107**	1.214*	1.046	1.005	0.768*
Laguerre	Isomap	3	2	0.806*	**1.0**	0.236*	0.135*	2.24	1.179*	1.013	0.884*
Torus	PCA	3	3	0.946	**1.0**	0.167*	**0.107**	1.582*	1.062*	0.859*	0.764*
Torus	UMAP	3	3	0.92	0.965*	0.145*	0.124*	1.495*	**1.002**	0.868*	0.8*

Table 11.2: GPDMM parameter comparison for top-performing variations of the numbers of layers, order of the dynamics, and back-constraints (BCs).

			F1		Distance		Damping		LDJ	
Layers	**Order**	**BC**	**BM**	**CMU**	**BM**	**CMU**	**BM**	**CMU**	**BM**	**CMU**
1	1	None	0.941	**1.0**	**0.152**	**0.106**	**1.104**	1.041	0.943	0.769
1	2	None	0.918	**1.0**	0.169	0.108	1.108	1.055	0.936	0.788
2	1	None	0.945	**1.0**	0.17	0.107	1.331*	**1.026**	0.914	0.857
1	1	Kernel	0.924	**1.0**	0.167	0.108	1.169	1.073	0.928	0.78
1	1	GP	0.935	**1.0**	0.171	0.108	1.369	1.066	**0.949**	0.784
1	1	Linear	**0.963**	**1.0**	0.171	0.145*	1.88*	2.213*	0.77*	0.782
1	1	MLP	0.934	**1.0**	0.212*	0.176*	1.752	11.658	0.881	0.766
1	1	C-Kernel	0.348*	0.261*	0.448*	0.331*	1.883*	3.776	0.992	**0.884**

illustrates how adding geometric and dimensionally reduced (DR) features to the latent space initialization impacts GPDMM performance. The table shows only top-performing configurations from a much larger set of parameters tested, including additional geometries and DR methods (see chapter A, tables A.1 and A.2 for expanded results). All geometries were constructed as described in section 11.4.2. The ellipse and torus were created using their parametric equations. The Chebyshev and Laguerre geometries were defined analogously to the example Fourier geometry in eq. equation 11.4. The column labeled "dims" indicates how many latent dimensions a given geometry occupied. Function-based geometries (e.g., Fourier), which lack a fixed dimensionality, were optimized as hyperparameters. From these experiments, we concluded that combining a Fourier geometry with a form of PCA produced the best set of results, particularly due to the consistently strong damping scores on both data sets compared to other approaches.

Table 11.1 also emphasizes the importance of the damping metric. Many variations of the GPDMM and our benchmarks generated movements that scored well on distance and LDJ, but when evaluated by eye, were severely under-expressing their range of motion or halting in place. Damping scores above around 1.3 were

Table 11.3: Comparison between the GDMM and benchmark models.

Model	F1		Distance		Damping		LDJ	
	BM	CMU	BM	CMU	BM	CMU	BM	CMU
GPDMM	0.93	**1.0**	**0.155**	**0.108**	**1.101**	**1.032**	**0.945**	0.77*
Sparse GPDMM	**0.936**	**1.0**	0.191*	0.113*	1.882*	1.145*	0.807*	0.726*
VAE	0.874*	**1.0**	0.189*	0.131*	1.328*	1.037	1.262*	1.228*
Transformer	0.903*	0.944*	0.356*	0.254*	3.318*	4.642*	0.854*	0.761*
LSTM	0.534*	0.871*	0.292*	0.206*	345.211	548.459*	0.818*	**0.886**

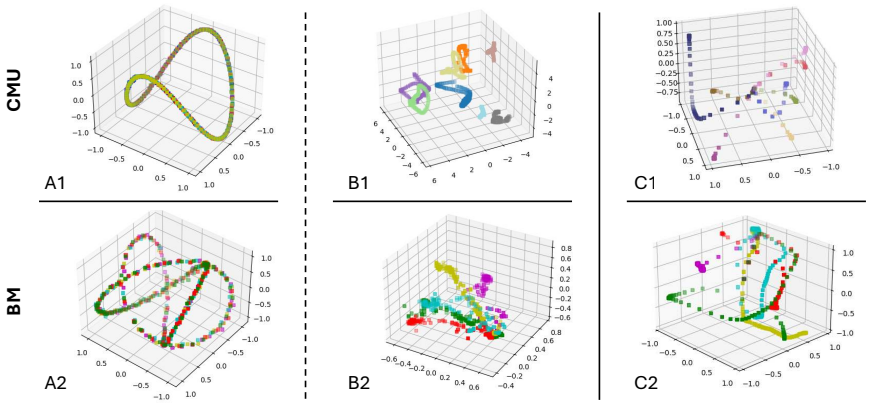


Figure 11.2: *GPDMM and LSTM Latent Space Visualizations.* The figure displays latent spaces for select feature dimensions of top-performing GPDMMs and LSTMs. The top plots (1) display models trained on the CMU data set, while the bottom plots (2) show the BM data set. The left-hand and middle plots (A and B) depict GPDMM latent spaces. "A" plots illustrate the fourier basis geometries. "B" plots present latent features that were initialized by dimensionally reduced representations of the data, "B1" with standard PCA and "B2" with RBF kernel PCA. The right-hand plots (C) show the first three dimensions of the LSTM's hidden state dynamics.

found to be indicative of poor performance. Furthermore, in row 4 (None; Random), the GPDMM scores well on LDJ, but very poorly on damping, indicating an artificially good LDJ score (further examination revealed the generated motion to be both under-expressed and noisy).

Table 11.2 reports the GPDMM's performance under variations in the layer count, order of the dynamical model, and type of back-constraint (BCs). The first row presents the GPDMM's performances with the Fourier-PCA latent spaces from table 11.1. The subsequent rows detail our most successful models optimized over hyperparameters (e.g. latent dimensions, geometries, and DRs) for specified numbers of layers, orders, and BCs. The terms "GP," "MLP," and "C-Kernel" respectively refer to Gaussian process, multi-layer perceptron, and circular kernel BCs. Based on these findings, we concluded that a simplified single-layer, first-order-dynamics model without BCs performed best.

Table 11.3 compares the GPDMM and its sparse variant ($M = N/2$) against our three benchmark models: the VAE, LSTM, and transformer. On the BM data set, the GPDMM maintained strong performance across all metrics, significantly outperforming the other approaches in distance, LDJ, and damping, and performing within significance of the highest F1 score produced by the sparse model. Performances on the CMU data set closely matched the BM data set. Again, the GPDMM performed well on all metrics, scoring significantly above the other models on distance. It tied with the sparse model, the transformer, and the VAE for the best F1 score, and scored within significance of the VAE on damping. While the LSTM achieved the top LDJ value, its extremely high damping makes this score unreliable.

Figure 2 presents feature dimensions of the GPDMM's latent space and the LSTM's hidden state dynamics when separately trained on the CMU (the top figures marked with 1) and BM (bottom, marked 2) data sets. Plots A show the GPDMM's geometric representations for capturing dynamics. A1 shows the first three Fourier basis functions (eq. 11.4), and A2 shows functions 4-6. Different function sets were selected for each plot as the models produce visually similar latent representations for identical functions. Data points are aligned with the geometry and distributed sequentially across its breadth to promote accurate state

transitions (see section 11.4.2). Plots B show features optimized for class discrimination and exhibit more variation across the space. Plots C display the first three dimensions of the LSTM's hidden state dynamics, combining information on dynamics and class discrimination together. B2, C2, and C1 display a divergence from a common centerpoint. This divergent pattern accurately represents the movement structure in the BM dataset but not in the CMU dataset. The GPDMM approach correctly captures this distinction (B1), while the LSTM misrepresents it (C1).

11.6 Discussion

We introduced the GPDMM, a mixture-of-experts framework that leverages geometry-embedded latent spaces to classify and generate diverse motion classes from minimal data. Across two human-motion datasets, our model consistently outperformed or matched popular neural baselines, while preserving the interpretability inherent to GPDMs. These results highlight the ability of the GPDMM to produce robust, data-efficient motion analysis and generation in scenarios such as prosthetic control, where patient-specific data sets are limited in size, and reliability and transparency are critical.

11.7 Acknowledgements

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Jesse St. Amand. This research was funded through the European Research Council ERC 2019-SYG under EU Horizon 2020 research and innovation programme (grant agreement No. 856495, RELEVANCE). The CMU mocap data set used in this project was obtained from mocap.cs.cmu.edu and was created with funding from NSF EIA-0196217.

Part IV.

Conclusion & Future Directions

12 Conclusion

The work presented in this thesis represents a systematic progression toward solving fundamental challenges in human motion modeling, culminating in the development of the GPDMM. Beginning with the application of hierarchical Gaussian process models to prosthetic control, continuing through the development of Gaussian process back-constraints and a novel feature selection technique using the information bottleneck method, and concluding with the comprehensive evaluation of the GPDMM framework. This research demonstrates how interpretable probabilistic approaches, like GPs, can effectively address the complexities of human movement representation when learning from small data sets.

Our initial investigations with hierarchical GPDMs in prosthetic applications revealed both the potential and limitations of task-specific approaches. While successful in modeling individual movement types, these early implementations highlighted the need for frameworks capable of handling multiple movement classes while maintaining reliable forward prediction. This insight drove the development of mixture model approaches and the exploration of methods for optimizing latent space organization.

The integration of information bottleneck methods for feature selection further refined our understanding of efficient movement representation, providing principled approaches for identifying and organizing the most informative aspects of movement data. This work established theoretical foundations for optimizing latent space organization, complementing our empirical findings regarding geometric initialization.

The subsequent investigation of back-constraints and geometric latent space features proved pivotal in understanding how to effectively organize movement representations. Contrary to initial expectations, we found that careful initialization of latent spaces with appropriate geometric features outperformed explicit topological constraints. This discovery led to significant simplifications in model architecture while improving performance, demonstrating how practical insights can lead to more elegant and effective solutions.

These developments culminated in the GPDMM, which synthesizes these insights into a unified framework capable of addressing multiple movement modeling challenges simultaneously. The model's success in single-example learning scenarios validates the fundamental premise that probabilistic methods with appropriate structural priors can effectively compensate for limited training data.

The progression of this research also highlights the value of maintaining interpretability in motion modeling systems. From our initial work with prosthetic control through to the final GPDMM implementation, the ability to understand and visualize model behavior has remained central to our approach. This emphasis on interpretability, combined with real-time computational capability, ensures that these theoretical advances can translate effectively to practical applications.

Taken together, this body of work demonstrates that principled probabilistic approaches to motion modeling, when combined with careful consideration of prior assumptions about the data space and model architecture, can achieve state-of-the-art performance while maintaining interpretability and computational efficiency. These results suggest promising directions for future research in both theoretical development and practical applications of motion modeling systems.

13 Future Directions

The successful development of the GPDMM framework opens several promising avenues for both theoretical advancement and practical applications. These directions span from immediate technical improvements to broader applications in clinical settings, with prosthetic control representing a particularly promising domain for future development.

13.0.1 Prosthetic Control Applications

Building on our initial work combining EMG signals with motion capture data for hand movement prediction (see Taubert et al. [104] or section 7), prosthetic applications present a natural extension for the GPDMM framework. The framework's ability to classify and generate movements from partial observations could enable more intuitive control systems, while its capacity to learn from limited examples makes it especially valuable for developing personalized control strategies. Our bimanual movement data set provides an ideal foundation for investigating coordinated movement prediction, particularly in applications where one arm's movements might inform control strategies for the other. This research direction could benefit patients who retain function in one arm but have impaired control of the other.

Preliminary results in prosthetic control simulations have revealed interesting trade-offs between GPDMs and GPDMMs. While GPDMMs excel at movement classification in their standard form, GPDMs enhanced with a feedback routine have shown promising results, able to implicitly classify movements through the generative process while maintaining smooth switching between movement types. This suggests that single GPDMs may have advantages over GPDMMs in this context. It may also imply the benefits of a hierarchical model, that utilizes one overarching dynamical GP for task selection and switching, and specialized dynamical GPs for fine-tuned control.

13.0.2 Model Architecture and Learning

The comparison between GPDMMs and GPDMs with style variables (the goal coordination method used in Taubert et al. [104], section 7) presents an important research direction. Style variables traditionally serve to separate movement classes through the GP emission kernel, with applications ranging from distinguishing movement styles (such as the enthusiasm or stiffness of a handshake [103]) to enabling smooth transitions between these styles [102]. Our work has demonstrated that this framework can be extended beyond style separation to more general applications, such as goal prediction in reach-to-grasp movements. Future research should investigate the trade-offs between these approaches, particularly their effectiveness in coordinating across different movement classes. The integration of style variables into the GPDMM framework could provide more nuanced control over latent space configuration and enable smoother transitions between movements by gradually blending them during the switching period.

The development of more sophisticated sparse approximation methods represents an avenue for enhancing the GPDMM's scalability. Our current implementation successfully employs FITC approximations, but future work should explore alternative approaches. One promising direction is variational sparse GPs [26, 106], which approximate the posterior distribution over the inducing points. Another approach is structured kernel interpolation, which leverages a regular grid of inducing points to enable efficient matrix-vector multiplication. For stationary kernels, this can be particularly efficient through the use of the Fast Fourier Transform [115]. These methods could help identify the optimal trade-offs between data size, accuracy, and computational efficiency for GPDMMs, while systematic comparisons with neural networks could reveal the data regimes where each approach is most effective.

13.0.3 Adaptation and Transfer Learning

Transfer learning approaches could enable trained models to adapt to new movement types or to individual variations between users. One key area for investigation is domain adaptation techniques that can learn invariant features across domains, adapting the model's parameters to new contexts while preserving its core movement representations. These techniques can bridge the gap between different users' movement patterns, sensor configurations, or environmental conditions. Another important direction is online learning methods for continuous adaptation, which would enable the model to incrementally learn new movements and adapt to specific users without requiring complete retraining. These capabilities would be particularly valuable in prosthetic applications, where the model's ability to adapt to new users or conditions could enable better personalization and expand the range of possible applications.

13.0.4 Multi-Modal Integration and Real-World Applications

The incorporation of diverse sensor modalities represents another direction for development. While our initial work successfully integrated EMG signals with motion capture data, future work should explore optimal fusion strategies for different sensor combinations, real-time processing requirements and latency considerations, and robustness to sensor noise and missing data. High-resolution infrared motion capture systems provide detailed movement data with many degrees of freedom, ideal for establishing comprehensive movement models. However, accelerometers and other wearable sensors offer more practical solutions for real-world applications. Future work should develop methods to leverage high-resolution motion capture data for model development while enabling effective control using more practical sensor configurations.

13.0.5 Extended Temporal Modeling

The ability to plan and execute sequences of movements in real-time presents significant challenges for movement prediction systems. Future research should address how to better model both immediate movement dynamics and longer-term planning, enabling the GPDMM to generate contextually appropriate sequences of movements. This could involve developing hierarchical models that operate at multiple timescales, from fine-grained movement execution to higher-level task planning. Such models could integrate with cognitive theories of movement planning and adapt in real-time to changing goals and environmental conditions.

13.0.6 Final Remarks

The directions outlined here represent both immediate technical challenges and longer-term research opportunities. From improving model scalability and adaptation to developing more sophisticated movement planning systems, these advances could significantly enhance the practical utility of movement prediction systems in rehabilitation and assistive technologies. The GPDMM, as a flexible and scalable probabilistic framework, provides a solid foundation for addressing these challenges and for expanding into a broader range of applications than those explored in this thesis.

Part V.

Appendix

Single-Example Learning in a Mixture of GPDMs with Latent Geometries

A.

Tables A.1 and A.2 provide expanded results for table 11.1 in section 11.4.2, respectively on the BM and CMU data sets. As in table 11.1, the best performing model per metric is highlighted in bold. The column headers "Geo", "Dims", and "DR" represent the geometry type, dimensionality of the geometry, and dimensionality reduction method, respectively. In the Dims column, "NA" (not applicable) indicates latent spaces without geometric features, while "all" indicates that geometric features occupied the total number of dimensions in the latent space.

Rows are arranged in descending order of a cumulative score (Cumul.) that is computed as the product of the distance (Dist.), the damping (Damp.), and the inverse F1 metrics. The log dimensionless jerk (LDJ) was excluded from this calculation because it fell into a broad range for fitness, with good scores typically between $0.7 - 1.0$. The cumulative score was used to select the best performing model among alternative parameterizations—such as the dimensionality of the geometry (Dims)—for latent initializations of specific geometries (Geo) and dimensionality reduction (DR) methods. This score serves as a quick reference for comparing relative performances.

The Geo column uses the following terms and abbreviations. Fourier, Chebyshev, Legendre, Laguerre, Walsh, Zernike, and spherical harmonics (SHs) refer to basis functions used to construct their respective geometries (see section 11.4.2). Random sine waves (RSWs) represents an alternative to the Fourier geometry, using sine functions with random frequencies, amplitudes, and phases instead of set values. Ellipse, torus, toroid, sphere, klein bottle (KB), and möbius strip (MS) refer to geometries constructed from their corresponding parametric equations. Linear refers to geometries made up of equal interval lines. The concentric circles (CCs) geometry represents a unique latent space structure of multiple circles arranged concentrically, where each circle represents a different movement class.

The DR column uses the following terms and abbreviations. kPCA refers to radial basis function kernel principal component analysis. UMAP (uniform manifold approximation and projection) appears in two forms: C and E, where C indicates construction using cosine distance and E indicates euclidean distance.

Here we distinguish between a toroid and torus. Both are constructed from the parametric equations of a torus:

$$\begin{aligned} \mathbf{x}(\boldsymbol{\theta}) &= (R + r\cos n\boldsymbol{\theta})\cos\boldsymbol{\phi}, \\ \mathbf{y}(\boldsymbol{\theta}) &= (R + r\cos n\boldsymbol{\theta})\sin\boldsymbol{\phi}, \\ \mathbf{z}(\boldsymbol{\theta}) &= r\sin n\boldsymbol{\theta}, \end{aligned} \tag{A.1}$$

where R defines the outer radius of the torus, r the inner radius, and n represents the number of loops around the inner radius. Both geometries set $\boldsymbol{\theta}$ to the vector defined in section 11.4.2. The key difference is that the torus geometry sets $\boldsymbol{\phi}$ to an equal interval range from 0 to $2\pi q$ (where q is a hyperparameter), while the toroid sets $\boldsymbol{\phi} = \boldsymbol{\theta}$. Due to the toroid's superior performance, we included it in the main text under the label "torus".

Three geometries that stand out in tables A.1 and A.2 but were excluded from the main text are the KB, CCs, and RSWs.

The KB geometry was excluded because it belongs to the same category as the ellipse and torus—geometries defined by parametric equations. Both ellipse and torus performed comparably to the KB while being more intuitive to understand.

The CCs geometry was likewise excluded despite outperforming both the ellipse and torus because it was less intuitive than other geometries and still underperformed compared to the Fourier geometry. Future work may benefit from exploring which aspects of the CCs geometry enabled its strong performance and whether these can be applied to other geometries. For instance, CCs uniquely utilized among the other geometries a

discrete term that separated the circles into different regions of the same dimensional space. I would suggest applying a similar tactic to other geometries to see if this improves performance more generally.

RSWs were excluded from the main text due to their similarity to the Fourier geometry, which achieved better overall performance when optimized for the number of Dims and the DR method. Notably, on both data sets, the best performing RSW models used "pure" geometric configurations (without DR features), outperforming comparable "pure" Fourier geometries. This suggests that randomizing sine wave parameters provides greater flexibility in the latent space to simultaneously model dynamics and classification, whereas the optimal Fourier approach benefited from combining geometric and DR features to separately model dynamics and classification, respectively.

All other geometries represented exclusively in these tables were excluded from the main text due to space constraints and inferior performances.

Table A.1: A comparison on the BM dataset of top-performing variations of the GPDMM initialized with different latent geometries and dimensionality reduction methods. See text for details including an explanation of terms.

Geo	Dims	DR	F1	Dist.	Damp.	LDJ	Cumul.
Fourier	11	kPCA	0.941*	0.152*	**1.104**	0.943*	0.178
RSWs	All	None	0.928*	0.142	1.195	0.844*	0.183
Fourier	8	UMAP-C	0.905*	0.142	1.211	0.891*	0.19
Fourier	9	UMAP-E	0.89*	0.14	1.231	0.906*	0.194
CCs	2	kPCA	0.931*	0.147*	1.267*	0.845*	0.2
CCs	2	UMAP-E	0.929*	0.148	1.358*	0.89*	0.216
CCs	2	UMAP-C	0.9*	0.151*	1.336*	0.883*	0.224
Fourier	All	None	0.938*	**0.131**	1.624*	0.807*	0.227
Fourier	13	PCA	0.928*	0.174*	1.214*	1.005	0.228
Toroid	3	UMAP-E	0.92*	0.145*	1.495*	0.868*	0.236
KB	3	kPCA	0.963	0.153*	1.504*	0.787*	0.239
Ellipse	2	kPCA	0.955	0.151*	1.526*	0.775*	0.241
Toroid	3	UMAP-C	0.92*	0.145	1.558*	0.85*	0.246
Toroid	3	kPCA	0.932*	0.152*	1.513*	0.789*	0.247
Chebyshev	All	None	0.943*	0.145	1.625*	0.813*	0.25
KB	3	UMAP-C	0.92*	0.141	1.677*	0.84*	0.257
Sphere	3	UMAP-C	0.905*	0.164*	1.489*	1.023	0.27
CCs	2	PCA	0.941*	0.166*	1.531*	0.862*	0.27
KB	3	UMAP-E	0.901*	0.143	1.721*	0.862*	0.273
Laguerre	All	None	0.911*	0.152	1.639*	0.73*	0.273
Toroid	3	PCA	0.946*	0.167*	1.582*	0.859*	0.279
Sphere	3	UMAP-E	0.902*	0.165*	1.557*	1.026	0.285
KB	3	PCA	0.937*	0.167*	1.608*	0.858*	0.287
Laguerre	3	UMAP-C	0.903*	0.147	1.784*	0.873*	0.29
RSWs	3	PCA	0.955	0.169*	1.65*	0.861*	0.292
Laguerre	4	UMAP-E	0.915*	0.146	1.864*	0.869*	0.297
Torus	3	UMAP-C	0.874*	0.156	1.673*	1.16*	0.299
MS	2	UMAP-E	0.913*	0.152	1.819*	0.858*	0.303
Sphere	3	PCA	**0.98**	0.182*	1.633*	0.947*	0.303
Ellipse	2	PCA	0.937*	0.172*	1.657*	0.87*	0.304
Torus	3	UMAP-E	0.907*	0.166	1.668*	1.139*	0.305
MS	2	PCA	0.959	0.169*	1.736*	0.866*	0.306
Ellipse	2	UMAP-E	0.92*	0.145	1.978*	0.86*	0.312
Ellipse	2	UMAP-C	0.91*	0.15	1.902*	0.843*	0.314
Laguerre	5	kPCA	0.939*	0.161*	1.834*	0.73*	0.314
Chebyshev	5	kPCA	0.93*	0.165*	1.791*	0.728*	0.318
RSWs	5	UMAP-C	0.92*	0.163*	1.916*	0.898*	0.339
Torus	3	PCA	0.898*	0.176*	1.783*	1.047*	0.349
Laguerre	3	PCA	0.938*	0.175*	1.916*	0.863*	0.357
None	NA	PCA	0.941*	0.175*	1.925*	0.882*	0.358
Legendre	All	None	0.876*	0.176*	1.833*	0.75*	0.368
Chebyshev	2	PCA	0.942*	0.179*	1.936*	0.877*	0.368
Legendre	2	PCA	0.942*	0.179*	1.936*	0.877*	0.368
Zernike	All	None	0.94	0.243*	1.474*	1.125*	0.381
Linear	3	PCA	0.937*	0.182*	2.013*	0.877*	0.391
Legendre	5	kPCA	0.919*	0.181*	2.022*	0.729*	0.398
Torus	3	kPCA	0.82*	0.169*	1.939*	1.109*	0.4

Continued on next page

Table A.1 – continued from previous page

Geo	Dims	DR	F1	Dist.	Damp.	LDJ	Cumul.
CCs	2	Isomap	0.762*	0.24*	1.319*	1.009	0.415
MS	2	kPCA	0.971	0.228*	1.798*	0.794*	0.422
RSWs	3	UMAP-E	0.911*	0.156*	2.538*	0.906*	0.435
MS	2	UMAP-C	0.915*	0.156*	2.597	0.833*	0.443
Fourier	4	Isomap	0.778*	0.193*	1.864	1.019	0.462
Chebyshev	5	UMAP-C	0.93*	0.161*	2.728*	0.863*	0.472
Sphere	3	Isomap	0.806*	0.218*	1.761*	1.055*	0.476
Chebyshev	5	UMAP-E	0.92*	0.157	2.883*	0.841*	0.492
SHs	All	None	0.946*	0.292*	1.678*	1.119*	0.518
Ellipse	2	Isomap	0.824*	0.216*	1.991*	1.05	0.522
RSWs	5	Isomap	0.816*	0.206*	2.131	1.02	0.538
Linear	3	kPCA	0.955*	0.209*	2.482*	0.744*	0.543
Legendre	4	UMAP-E	0.907*	0.163	3.197*	0.878*	0.575
Linear	3	UMAP-C	0.915*	0.163	3.407*	0.855*	0.607
Legendre	5	UMAP-C	0.916*	0.165*	3.511*	0.864*	0.632
None	NA	UMAP-C	0.909*	0.17*	3.401*	0.864*	0.636
Laguerre	3	Isomap	0.806*	0.236*	2.24	1.013	0.656
None	NA	UMAP-E	0.909*	0.172*	3.523*	0.881*	0.667
Linear	3	UMAP-E	0.894*	0.176*	3.574*	0.847*	0.704
Linear	3	Isomap	0.804*	0.228*	2.643*	1.034	0.75
KB	3	Isomap	0.812*	0.219*	2.973*	1.047	0.802
Sphere	3	kPCA	0.951*	0.258*	3.104*	1.106*	0.842
Toroid	3	Isomap	0.806*	0.222*	3.063	1.037	0.844
Torus	3	Isomap	0.807*	0.242*	2.936*	1.175*	0.88
RSWs	5	kPCA	**0.984**	0.266*	3.51*	0.861*	0.949
MS	2	Isomap	0.798*	0.234*	3.241*	1.039	0.95
Legendre	3	Isomap	0.817*	0.238*	3.591	1.019	1.046
None	NA	Isomap	0.781*	0.233*	3.551	1.026	1.059
Chebyshev	2	Isomap	0.807*	0.249*	3.866	1.043	1.193
Walsh	All	None	0.889*	0.256*	4.871*	0.963*	1.403
None	NA	kPCA	0.959	0.232*	6.44*	0.784*	1.558
Linear	All	None	0.862*	0.295*	34.242	0.766*	11.719
Random	All	None	0.829*	0.359*	94.513*	**0.998**	40.929

Table A.2: A comparison on the CMU dataset of top-performing variations of the GPDMM initialized with different latent geometries and dimensionality reduction methods. See text for details including an explanation of terms.

Geo	Dims	DR	F1	Dist.	Damp.	LDJ	Cumul.
Fourier	5	PCA	**1.0**	0.107	1.04	0.79*	0.111
Toroid	3	PCA	**1.0**	0.107	1.062*	0.764*	0.114
CCs	2	PCA	**1.0**	0.108	1.057*	0.787*	0.114
Fourier	4	kPCA	**1.0**	0.117	**0.99**	0.82*	0.116
CCs	2	UMAP-C	**1.0**	0.117	**0.992**	0.813*	0.116
CCs	2	kPCA	**1.0**	0.114	1.022	0.775*	0.117
Toroid	3	UMAP-C	**1.0**	0.114	1.027	0.8*	0.117
Toroid	3	kPCA	**1.0**	0.115	1.025	0.774*	0.118
CCs	2	Isomap	**1.0**	0.115	1.061*	0.86*	0.122
CCs	2	UMAP-E	0.983*	0.123	0.973	0.812*	0.122
Fourier	7	UMAP-C	0.991	0.122	1.007	0.846*	0.124
KB	3	kPCA	**1.0**	0.12	1.04*	0.752*	0.125
Ellipse	2	kPCA	**1.0**	0.116	1.084*	0.718*	0.126
Ellipse	2	UMAP-C	**1.0**	0.117	1.084*	0.791*	0.127
KB	3	UMAP-C	**1.0**	0.117	1.091*	0.797*	0.128
KB	3	PCA	**1.0**	0.117	1.093*	0.764*	0.128
Ellipse	2	PCA	**1.0**	0.118	1.095*	0.764*	0.129
Toroid	3	UMAP-E	0.965*	0.124	**1.002**	0.8*	0.129
Fourier	5	Isomap	0.996	0.122	1.111*	0.844*	0.136
Fourier	2	UMAP-E	0.987	0.123	1.09*	0.782*	0.136
KB	3	UMAP-E	0.991	0.125	1.076*	0.783*	0.136
RSWs	All	None	**1.0**	0.124	1.106	0.81*	0.137
Toroid	3	Isomap	**1.0**	0.125	1.13*	0.882*	0.141
Ellipse	2	UMAP-E	0.987	0.125	1.122*	0.795*	0.142
Fourier	All	None	**1.0**	0.11	1.297*	0.77*	0.143
RSWs	5	Isomap	**1.0**	0.125	1.149*	0.844*	0.144
KB	3	Isomap	**1.0**	0.132	1.129*	0.876*	0.149
Ellipse	2	Isomap	0.996	0.134	1.148*	0.881*	0.154
Sphere	3	Isomap	**1.0**	0.128	1.23*	0.879*	0.157
MS	2	Isomap	**1.0**	0.134	1.181*	0.864*	0.158
Sphere	3	UMAP-C	**1.0**	0.129	1.233*	0.894*	0.159
Laguerre	2	Isomap	**1.0**	0.135	1.179*	0.884*	0.159
Linear	3	UMAP-C	**1.0**	0.13	1.236*	0.779*	0.161
Linear	3	Isomap	**1.0**	0.135	1.199*	0.891*	0.162
Chebyshev	3	UMAP-C	**1.0**	0.13	1.253*	0.775*	0.163
RSWs	3	UMAP-C	**1.0**	0.132	1.235*	0.837*	0.163
MS	2	UMAP-C	**1.0**	0.125	1.321*	0.805*	0.165
Legendre	5	Isomap	**1.0**	0.138	1.193*	0.876*	0.165
None	NA	UMAP-C	0.991	0.126	1.309*	0.784*	0.166
Legendre	3	UMAP-C	**1.0**	0.129	1.284*	0.779*	0.166
Chebyshev	2	Isomap	0.996	0.138	1.201*	0.895*	0.166
Laguerre	3	UMAP-C	**1.0**	0.128	1.305*	0.777*	0.167
MS	2	PCA	**1.0**	0.127	1.326*	0.763*	0.168
MS	2	UMAP-E	0.987	0.134	1.313	0.797*	0.178
Laguerre	3	UMAP-E	0.991	0.137	1.353*	0.777*	0.187
None	NA	Isomap	0.996	0.135	1.398	0.902*	0.189
Chebyshev	5	UMAP-E	0.987	0.14	1.338*	0.759*	0.19

Continued on next page

Table A.2 – continued from previous page

Geo	Dims	DR	F1	Dist.	Damp.	LDJ	Cumul.
Laguerre	3	PCA	**1.0**	0.139	1.472*	0.748*	0.205
Legendre	4	UMAP-E	0.996	0.144	1.435*	0.759*	0.207
MS	2	kPCA	**1.0**	0.155	1.364*	0.778*	0.211
RSWs	5	UMAP-E	0.987	0.143	1.503	0.84*	0.218
Sphere	3	UMAP-E	0.991	0.143	1.546*	0.913*	0.223
Chebyshev	All	None	**1.0**	0.126	1.782*	0.816*	0.225
None	NA	UMAP-E	0.991	0.136	1.758*	0.784*	0.241
Walsh	All	None	**1.0**	0.138	1.86*	0.688*	0.257
Laguerre	All	None	**1.0**	0.136	1.915*	0.784*	0.26
Chebyshev	4	PCA	**1.0**	0.143	1.923*	0.751*	0.275
RSWs	3	PCA	**1.0**	0.146	1.985*	0.825*	0.29
Linear	3	UMAP-E	0.996	0.144	2.201	0.773*	0.318
Legendre	5	PCA	**1.0**	0.152*	2.127*	0.75*	0.323
Laguerre	5	kPCA	**1.0**	0.145	2.382*	0.701*	0.345
Chebyshev	5	kPCA	**1.0**	0.15	2.764*	0.705*	0.415
None	NA	PCA	**1.0**	0.148	3.551*	0.785*	0.526
Linear	3	PCA	**1.0**	0.151	3.69*	0.755*	0.557
Legendre	All	None	**1.0**	0.154	3.657*	0.799*	0.563
Sphere	3	PCA	**1.0**	0.158	3.581*	0.917*	0.566
Torus	3	UMAP-C	0.309*	0.122	1.439*	1.09*	0.568
Torus	3	UMAP-E	0.296*	0.135	1.37*	1.047	0.625
Torus	3	Isomap	0.207*	0.13	1.309*	1.037	0.822
RSWs	3	kPCA	**1.0**	0.182	5.387*	0.926*	0.98
Torus	3	PCA	0.215*	0.142	1.972*	1.079*	1.302
Legendre	5	kPCA	**1.0**	0.173	9.313*	0.728*	1.611
Sphere	3	kPCA	**1.0**	0.177	9.505*	1.023	1.682
Torus	3	kPCA	0.113*	**0.091**	2.195*	1.144*	1.768
Zernike	All	None	0.996	0.182	12.357*	0.921*	2.258
Linear	3	kPCA	**1.0**	0.182	30.876*	0.739*	5.619
Linear	All	None	**1.0**	0.183	42.503*	0.849*	7.778
None	NA	kPCA	**1.0**	0.181	43.719	0.754*	7.913
Random	All	None	0.769*	0.338	37.257*	**0.992**	16.376
SHs	All	None	**1.0**	6.197e12	1.293*	1.025	8.013e12

Bibliography

[1] M. Al Borno, J. L. Hicks, and S. L. Delp. "The Effects of Motor Modularity on Performance, Learning and Generalizability in Upper-Extremity Reaching: A Computational Analysis". *Journal of The Royal Society Interface* 17.167 (2020), page 20200011.

[2] M. Althoff, M. Mayer, and R. Müller. "Automatic Synthesis of Human Motion from Temporal Logic Specifications". *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pages 4040–4046.

[3] Anthropic. "Claude 3.7 Sonnet". 2025.

[4] S. Balasubramanian, A. Melendez-Calderon, and E. Burdet. "A Robust and Sensitive Metric for Quantifying Movement Smoothness". *IEEE Transactions on Biomedical Engineering* 59.8 (2011), pages 2126–2136.

[5] S. Behnke and J. Stückler. "Hierarchical Reactive Control for Humanoid Soccer Robots". *International Journal of Humanoid Robotics* 5.3 (2008), pages 375–396.

[6] C. M. Bishop. "Pattern Recognition and Machine Learning". Volume 4. *Springer*, 2006.

[7] S. Bitzer and C. K. I. Williams. "Kick-Starting GPLVM Optimization via a Connection to Metric MDS". *Proceedings of the NIPS 2010 Workshop on Challenges of Data Visualization*.

[8] A. Blattmann, T. Dockhorn, S. Kulal, D. Mendelevitch, M. Kilian, D. Lorenz, Y. Levi, et al. "Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets". *arXiv preprint arXiv:2311.15127* (2023).

[9] A. Boukhayma, J.-S. Franco, and E. Boyer. "Surface Motion Capture Transfer with Gaussian Process Regression". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pages 184–192.

[10] E. B. Brokaw, I. Black, R. J. Holley, and P. S. Lum. "Hand Spring Operated Movement Enhancer (HandSOME): A Portable, Passive Hand Exoskeleton for Stroke Rehabilitation". *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 19.4 (2011), pages 391–399.

[11] A. Bruderlin and L. Williams. "Motion Signal Processing". *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '95. ACM Press*, 1995, pages 97–104.

[12] T. D. Bui, J. Yan, and R. E. Turner. "A Unifying Framework for Gaussian Process Pseudo-Point Approximations Using Power Expectation Propagation". *Journal of Machine Learning Research* 18.104 (2017), pages 1–72.

[13] T. D. Bui. "Efficient Deterministic Approximate Bayesian Inference for Gaussian Process Models". PhD thesis. University of Cambridge, 2018.

[14] A. Chadwell, L. Kenney, S. Thies, A. Galpin, and J. Head. "The Reality of Myoelectric Prostheses: Understanding What Makes These Devices Difficult for Some Users to Control". *Frontiers in Neurorobotics* 10 (2016), page 7.

[15] J. Chen, M. Kim, Y. Wang, and Q. Ji. "Switching Gaussian Process Dynamic Models for Simultaneous Composite Motion Tracking and Recognition". *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pages 2655–2662.

[16] Y. Chen, X. Wang, Y. Ye, and X. Sun. "An Explainable Machine Learning Framework for Lower Limb Exoskeleton Robot System". *Wireless Sensor Networks*. Edited by H. Ma, X. Wang, L. Cheng, L. Cui, L. Liu, and A. Zeng. Volume 1715. Communications in Computer and Information Science. Singapore: *Springer Nature Singapore*, 2022, pages 79–93.

[17] Z. Chen, H. Min, D. Wang, Z. Xia, F. Sun, and B. Fang. "A Review of Myoelectric Control for Prosthetic Hand Manipulation". *Biomimetics* 8.3 (2023), page 328.

[18] E. Chiovetto, A. Salatiello, A. d'Avella, and M. A. Giese. "Toward a Unifying Framework for the Modeling and Identification of Motor Primitives". *Frontiers in Computational Neuroscience* 16 (2022), page 926345.

[19] C.-C. Chiu and S. Marsella. "Gesture Generation with Low-Dimensional Embeddings". *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*. 2014, pages 781–788.

[20] M. M. Churchland, J. P. Cunningham, M. T. Kaufman, J. D. Foster, P. Nuyujukian, S. I. Ryu, and K. V. Shenoy. "Neural Population Dynamics during Reaching". *Nature* 487.7405 (2012), pages 51–56.

[21] CMU Graphics Lab. "Carnegie Mellon University Motion Capture Database". URL: http://mocap.cs.cmu.edu.

[22] R. Dabral, M. H. Mughal, V. Golyanik, and C. Theobalt. "Mofusion: A Framework for Denoising-Diffusion-Based Motion Synthesis". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pages 9760–9770.

[23] A. Damianou. "Deep Gaussian Processes and Variational Propagation of Uncertainty". PhD thesis. University of Sheffield, 2015.

[24] A. Damianou and N. D. Lawrence. "Deep Gaussian Processes". *Artificial Intelligence and Statistics*. 2013, pages 207–215.

[25] A. Damianou, M. Titsias, and N. Lawrence. "Variational Gaussian Process Dynamical Systems". *Advances in Neural Information Processing Systems*. Volume 24. 2011.

[26] A. C. Damianou, M. K. Titsias, and N. D. Lawrence. "Variational Inference for Latent Variables and Uncertain Inputs in Gaussian Processes". *Journal of Machine Learning Research* (2016).

[27] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. "The Helmholtz Machine". *Neural Computation* 7.5 (1995), pages 889–904.

[28] S. L. Delp, F. C. Anderson, A. S. Arnold, P. Loan, A. Habib, C. T. John, E. Guendelman, and D. G. Thelen. "OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement". *IEEE Transactions on Biomedical Engineering* 54.11 (2007), pages 1940–1950.

[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding". *arXiv preprint arXiv:1810.04805* (2018).

[30] E. Dorschky. "Optimal Control and Machine Learning in Biomechanics: Hybrid Models for Inertial Sensor-Based Gait Analysis and Virtual Footwear Testing". PhD thesis. Erlangen: Friedrich-Alexander-Universität Erlangen-Nürnberg, 2023.

[31] A. D. Dragan, K. C. Lee, and S. S. Srinivasa. "Legibility and Predictability of Robot Motion". *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE. 2013, pages 301–308.

[32] A. Driemel, S. Har-Peled, and C. Wenk. "Approximating the Fréchet Distance for Realistic Curves in near Linear Time". *Proceedings of the Twenty-Sixth Annual Symposium on Computational Geometry*. 2010, pages 365–374.

[33] S. Eleftheriadis, T. Nicholson, M. Deisenroth, and J. Hensman. "Identification of Gaussian Process State Space Models". *Advances in Neural Information Processing Systems*. Volume 30. 2017.

[34] D. Farina, I. Vujaklija, M. Sartori, T. Kapelner, F. Negro, N. Jiang, K. Bergmeister, A. Andalib, J. Principe, and O. C. Aszmann. "Man/Machine Interface Based on the Discharge Timings of Spinal Motor Neurons after Targeted Muscle Reinnervation". *Nature Biomedical Engineering* 1.2 (2017), page 0025.

[35] A. W. Franzke, M. B. Kristoffersen, R. M. Bongers, A. Murgia, B. Pobatschnig, F. Unglaube, and C. K. van der Sluis. "Users' and Therapists' Perceptions of Myoelectric Multi-Function Upper Limb Prostheses with Conventional and Pattern Recognition Control". *PloS One* 14.8 (2019), e0220899.

[36] R. Frigola. "Bayesian Time Series Learning with Gaussian Processes". PhD thesis. University of Cambridge, 2015.

[37] P. D. Ganzer, S. C. Colachis, M. A. Schwemmer, D. A. Friedenberg, C. F. Dunlap, C. E. Swiftney, A. F. Jacobowitz, D. J. Weber, M. A. Bockbrader, and G. Sharma. "Restoring the Sense of Touch Using a Sensorimotor Demultiplexing Neural Interface". *Cell* 181.4 (2020), pages 763–773.

[38] GPy. "GPy: A Gaussian Process Framework in Python". since 2012. 2012. URL: http://github.com/SheffieldML/GPy.

[39] F. S. Grassia. "Practical Parameterization of Rotations Using the Exponential Map". *Journal of Graphics Tools* 3.3 (1998), pages 29–48.

[40] R. Gray. "Changes in Movement Coordination Associated with Skill Acquisition in Baseball Batting: Freezing/Freeing Degrees of Freedom and Functional Variability". *Frontiers in Psychology* 11 (2020), page 1295.

[41] S. Grillner and P. Wallen. "Central Pattern Generators for Locomotion, with Special Reference to Vertebrates". *Annual Review of Neuroscience* (1985).

[42] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. "Style-Based Inverse Kinematics". *ACM SIGGRAPH 2004 Papers*. 2004, pages 522–531.

[43] J. M. Hahne, M. A. Schweisfurth, M. Koppe, and D. Farina. "Simultaneous Control of Multiple Functions of Bionic Hand Prostheses: Performance and Robustness in End Users". *Science Robotics* 3.19 (2018), eaat3630.

[44] E. Herrmann, H. Du, A. Antakli, D. Rubinstein, R. Schubotz, J. Sprenger, S. Hosseini, N. Cheema, I. Zinnikus, and M. Manns. "Motion Data and Model Management for Applied Statistical Motion Synthesis". *STAG*. 2019, pages 79–88.

[45] S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory". *Neural Computation* 9.8 (1997), pages 1735–1780.

[46] D. Holden, J. Saito, and T. Komura. "A Deep Learning Framework for Character Motion Synthesis and Editing". *ACM Transactions on Graphics* 35.4 (2016), pages 1–11.

[47] S. Ikemoto, H. Ben Amor, T. Minato, B. Jung, and H. Ishiguro. "Physical Human-Robot Interaction: Mutual Learning and Adaptation". *IEEE Robotics & Automation Magazine* 19.4 (2012), pages 24–35.

[48] J. M. Inouye and F. J. Valero-Cuevas. "Muscle Synergies Heavily Influence the Neural Control of Arm Endpoint Stiffness and Energy Consumption". *PLoS Computational Biology* 12.2 (2016), e1004737.

[49] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. "Adaptive Mixtures of Local Experts". *Neural Computation* 3.1 (1991), pages 79–87.

[50] S. Jain, Y. Ye, and C. K. Liu. "Optimization-Based Interactive Motion Synthesis". *ACM Transactions on Graphics* 28.1 (2009), pages 1–12.

[51] L. Jiang, Y. Wei, and H. Ni. "MotionPCM: Real-Time Motion Synthesis with Phased Consistency Model". *arXiv preprint arXiv:2501.19083* (2025).

[52] Y. Jiang. "Integrating Machine Learning and Physics Simulation Methodologies for Creating Digital Twins of Humans and Robots". PhD thesis. Stanford University, 2024.

[53] Z. Jiang, Y. Xie, J. Li, Y. Yuan, Y. Zhu, and Y. Zhu. "HARMON: Whole-Body Motion Generation of Humanoid Robots from Language Descriptions". *arXiv preprint arXiv:2410.12773* (2024).

[54] T. Kang, J. He, and S. I. H. Tillery. "Determining Natural Arm Configuration along a Reaching Trajectory". *Experimental Brain Research* 167.3 (2005), pages 352–361.

[55] Z. Kang, X. Wang, and Y. Mu. "BioMoDiffuse: Physics-Guided Biomechanical Diffusion for Controllable and Authentic Human Motion Synthesis". *arXiv preprint arXiv:2503.06151* (2025).

[56] A. A. Karim, T. Gaudin, A. Meyer, A. Buendia, and S. Bouakaz. "Procedural Locomotion of Multilegged Characters in Dynamic Environments". *Computer Animation and Virtual Worlds* 24.1 (2013), pages 3–15.

[57] O. Khatib, E. Demircan, V. De Sapio, L. Sentis, T. Besier, and S. Delp. "Robotics-Based Synthesis of Human Motion". *Journal of Physiology-Paris* 103.3-5 (2009), pages 211–219.

[58] D. P. Kingma and M. Welling. "Auto-Encoding Variational Bayes". *arXiv preprint arXiv:1312.6114* (2013).

[59] J. Ko and D. Fox. "GP-BayesFilters: Bayesian Filtering Using Gaussian Process Prediction and Observation Models". *Autonomous Robots* 27.1 (2009), pages 75–90.

[60] L. Kunze, O. Gunes, D. Hillier, M. Munks, H. Webb, P. Salvini, D. Omeiza, and M. Jirotka. "Towards Explainable and Trustworthy Collaborative Robots through Embodied Question Answering". 2022.

[61] M. L. Latash. "The Bliss (Not the Problem) of Motor Abundance (Not Redundancy)". *Experimental Brain Research* 217.1 (2012), pages 1–5.

[62] N. Lawrence. "Probabilistic Non-Linear Principal Component Analysis with Gaussian Process Latent Variable Models". *Journal of Machine Learning Research* 6.11 (2005).

[63] N. Lawrence, M. Rattray, and M. Titsias. "Efficient Sampling for Gaussian Process Inference Using Control Variables". *Advances in Neural Information Processing Systems*. Volume 21. 2008.

[64] N. D. Lawrence. "Learning for Larger Datasets with the Gaussian Process Latent Variable Model". *Artificial Intelligence and Statistics*. PMLR. 2007, pages 243–250.

[65] N. D. Lawrence. *Large Scale Learning with the Gaussian Process Latent Variable Model*. Technical report. University of Sheffield, 2008.

[66] N. D. Lawrence and J. Quiñonero-Candela. "Local Distance Preservation in the GP-LVM through Back Constraints". *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*. 2006, pages 513–520.

[67] O. Liu and D. Losey. "A Survey and Theoretical Analysis of Gaussian Process Latent Variable Models". *31st Conference on Neural Information Processing Systems (NIPS 2017)*. 2017.

[68] X. Liu, L. Yang, Z. Chen, J. Zhong, and F. Gao. "Motion Planning for a Legged Robot with Dynamic Characteristics". *Sensors* 24.18 (2024), page 6070.

[69] J. R. Magnus and H. Neudecker. "Matrix Differential Calculus with Applications in Statistics and Econometrics". *John Wiley & Sons*, 2019.

[70] E. V. Mascaro, Y. Yan, and D. Lee. "Robot Interaction Behavior Generation Based on Social Motion Forecasting for Human-Robot Interaction". *arXiv preprint arXiv:2402.04768* (2024).

[71] H. I. Masoud, Y. Zerehsaz, and J. J. Jin. "Analysis of Human Motion Variation Patterns Using UMPCA". *Applied Ergonomics* 59 (2017), pages 401–409.

[72] E. Medina, L. Loh, N. Gurung, K. H. Oh, and N. Heller. "Context-Based Interpretable Spatio-Temporal Graph Convolutional Network for Human Motion Forecasting". *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024, pages 3232–3241.

[73] V. Nair, J. Susskind, and G. E. Hinton. "Analysis-by-Synthesis by Learning to Invert Generative Black Boxes". *Artificial Neural Networks - ICANN 2008*. Edited by V. Kůrková, R. Neruda, and J. Koutník. Volume 5163. Lecture Notes in Computer Science. Berlin, Heidelberg: *Springer Berlin Heidelberg*, 2008, pages 971–981.

[74] N. Noceti, A. Sciutti, and F. Rea. "Modeling Human Motion: A Task at the Crossroads of Neuroscience, Computer Vision and Robotics". *Modelling Human Motion*. Edited by N. Noceti, A. Sciutti, and F. Rea. Cham: *Springer International Publishing*, 2020, pages 1–14.

[75] B. E. Olivas-Padilla, S. Manitsaris, and A. Glushkova. "Explainable AI in Human Motion: A Comprehensive Approach to Analysis, Modeling, and Generation". *Pattern Recognition* 151 (2024), page 110418.

[76] OpenAI. "GPT-o1". 2025.

[77] G. Palli, F. Ficuciello, U. Scarcia, C. Melchiorri, and B. Siciliano. "Experimental Evaluation of Synergy-Based in-Hand Manipulation". *IFAC Proceedings Volumes*. Volume 47. 3. 2014, pages 299–304.

[78] J. Pan, W. Li, L. Liu, K. Jia, T. Liu, and F. Chen. "Variable Selection Using Deep Variational Information Bottleneck with Drop-out-One Loss". *Applied Sciences* 13.5 (2023), page 3008.

[79] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga. "Pytorch: An Imperative Style, High-Performance Deep Learning Library". *Advances in Neural Information Processing Systems* 32 (2019).

[80] M. Perrone, S. P. Mell, J. T. Martin, S. J. Nho, S. Simmons, and P. Malloy. "Synthetic Data Generation in Motion Analysis: A Generative Deep Learning Framework". *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine* (2025).

[81] A. Phinyomark and E. Scheme. "EMG Pattern Recognition in the Era of Big Data and Deep Learning". *Big Data and Cognitive Computing* 2.3 (2018), page 21.

[82] A. D. Porzio and M. Coraggio. "A Personalized Data-Driven Generative Model of Human Motion". *arXiv preprint arXiv:2503.15225* (2025).

[83] J. Quinonero-Candela and C. E. Rasmussen. "A Unifying View of Sparse Approximate Gaussian Process Regression". *The Journal of Machine Learning Research* 6 (2005), pages 1939–1959.

[84] S. Raab, I. Leibovitch, P. Li, K. Aberman, O. Sorkine-Hornung, and D. Cohen-Or. "MoDi: Unconditional Motion Synthesis from Diverse Data". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pages 13873–13883.

[85] A. Ramadan, C. Boss, J. Choi, N. P. Reeves, J. Cholewicki, J. M. Popovich Jr, and C. J. Radcliffe. "Selecting Sensitive Parameter Subsets in Dynamical Models with Application to Biomechanical System Identification". *Journal of Biomechanical Engineering* 140.7 (2018), page 074503.

[86] D. Roetenberg, H. Luinge, and P. Slycke. *Xsens MVN: Full 6DOF Human Motion Tracking Using Miniature Inertial Sensors*. Tech. Rep 1. Xsens Motion Technologies BV, 2009, pages 1–7.

[87] M. J. Rosenstrauch and J. Krüger. "Safe Human-Robot-Collaboration-Introduction and Experiment Using ISO/TS 15066". *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*. IEEE. 2017, pages 740–744.

[88] C. Rudin. "Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead". *Nature Machine Intelligence* 1.5 (2019), pages 206–215.

[89] M. Sartori, G. Durandau, S. Došen, and D. Farina. "Robust Simultaneous Myoelectric Control of Multiple Degrees of Freedom in Wrist-Hand Prostheses by Real-Time Neuromusculoskeletal Modeling". *Journal of Neural Engineering* 15.6 (2018), page 066026.

[90] P. Schumacher, T. Geijtenbeek, V. Caggiano, V. Kumar, S. Schmitt, G. Martius, and D. F. Haeufle. "Emergence of Natural and Robust Bipedal Walking by Learning from Biologically Plausible Objectives". *iScience* (2025).

[91] L. Shi, Q. Liu, C. Zhou, W. Gao, H. Wu, Y. Zheng, and X. Li. "Uncovering the Secrets of Human-Like Movement: A Fresh Perspective on Motion Planning". *arXiv preprint arXiv:2409.10747* (2024).

[92] A. Simonetta and M. C. Paoletti. "ISO/IEC Standards and Design of an Artificial Intelligence System". 2024.

[93] E. Snelson and Z. Ghahramani. "Sparse Gaussian Processes Using Pseudo-Inputs". *Advances in Neural Information Processing Systems*. Volume 18. 2005.

[94] J. St. Amand and M. Giese. "Variable Selection in GPDMs Using the Information Bottleneck Method". *NeurIPS 2023 Workshop: Information-Theoretic Principles in Cognitive Systems*. 2023.

[95] J. St. Amand, L. Gizzi, and M. A. Giese. "Single-Example Learning in a Mixture of GPDMs with Latent Geometries". *arXiv preprint arXiv:2506.14563* (2025). arXiv: 2506.14563 [cs.LG].

[96] J. Starke and T. Asfour. "Kinematic Synergy Primitives for Human-Like Grasp Motion Generation". *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pages 4119–4125.

[97] M. Stelzer and O. von Stryk. "Applications of Efficient Forward Dynamics Simulation in Biomechanics". *Proceedings of the XVth International Conference on Mechanics in Medicine and Biology*. Singapore: *Citeseer*, 2006.

[98] N. Stergiou and L. M. Decker. "Human Movement Variability, Nonlinear Dynamics, and Pathology: Is There a Connection?" *Human Movement Science* 30.5 (2011), pages 869–888.

[99] S. Sun, G. De Araujo, J. Xu, S. Zhou, H. Zhang, Z. Huang, C. You, and X. Xie. "CoMA: Compositional Human Motion Generation with Multi-Modal Agents". *arXiv preprint arXiv:2412.07320* (2025).

[100] V. M. Systems. "Vicon Nexus (Version 2.8)". Computer software. Accessed 2021. 2021. URL: https://www.vicon.com/.

[101] V. M. Systems. "Vicon Shogun (Version 1.7)". Computer software. Accessed 2021. 2021. URL: https://www.vicon.com/.

[102] N. Taubert, A. Christensen, D. Endres, and M. A. Giese. "Online Simulation of Emotional Interactive Behaviors with Hierarchical Gaussian Process Dynamical Models". *Proceedings of the ACM Symposium on Applied Perception*. 2012, pages 25–32.

[103] N. Taubert, D. Endres, A. Christensen, and M. A. Giese. "Shaking Hands in Latent Space: Modeling Emotional Interactions with Gaussian Process Latent Variable Models". *KI 2011: Advances in Artificial Intelligence. Springer*, 2011, pages 330–334.

[104] N. Taubert, J. St. Amand, P. Kumar, L. Gizzi, and M. A. Giese. "Reactive Hand Movements from Arm Kinematics and EMG Signals Based on Hierarchical Gaussian Process Dynamical Models". *Artificial Neural Networks and Machine Learning – ICANN 2020. Springer*, 2020, pages 127–140.

[105] N. Tishby, F. C. Pereira, and W. Bialek. "The Information Bottleneck Method". *arXiv preprint physics/0004057* (2000).

[106] M. Titsias. "Variational Learning of Inducing Variables in Sparse Gaussian Processes". *Artificial Intelligence and Statistics*. PMLR. 2009, pages 567–574.

[107] M. Titsias and N. D. Lawrence. "Bayesian Gaussian Process Latent Variable Model". *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings. 2010, pages 844–851.

[108] R. Urtasun, D. J. Fleet, and N. D. Lawrence. "Modeling Human Locomotion with Topologically Constrained Latent Variable Models". *Human Motion – Understanding, Modeling, Capture and Animation. Springer*, 2007, pages 104–118.

[109] H. A. Varol, F. Sup, and M. Goldfarb. "Multiclass Real-Time Intent Recognition of a Powered Lower Limb Prosthesis". *IEEE Transactions on Biomedical Engineering* 57.3 (2009), pages 542–551.

[110] I. Vujaklija, V. Shalchyan, E. N. Kamavuako, N. Jiang, H. R. Marateb, and D. Farina. "Online Mapping of EMG Signals into Kinematics by Autoencoding". *Journal of NeuroEngineering and Rehabilitation* 15.1 (2018), page 21.

[111] J. Wang, A. Hertzmann, and D. J. Fleet. "Gaussian Process Dynamical Models". *Advances in Neural Information Processing Systems*. Volume 18. 2005.

[112] J. M. Wang, D. J. Fleet, and A. Hertzmann. "Gaussian Process Dynamical Models for Human Motion". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2 (2007), pages 283–298.

[113] J. M. Wang, D. J. Fleet, and A. Hertzmann. "Multifactor Gaussian Process Models for Style-Content Separation". *Proceedings of the 24th International Conference on Machine Learning*. 2007, pages 975–982.

[114] C. K. Williams and C. E. Rasmussen. "Gaussian Processes for Machine Learning". Volume 2. 3. *MIT press Cambridge, MA*, 2006.

[115] A. Wilson and H. Nickisch. "Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP)". *International Conference on Machine Learning. PMLR*, 2015, pages 1775–1784.

[116] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, and M. Funtowicz. "Transformers: State-of-the-Art Natural Language Processing". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 2020, pages 38–45.

[117] D. M. Wolpert and Z. Ghahramani. "Computational Principles of Movement Neuroscience". *Nature Neuroscience* 3.S11 (2000), pages 1212–1217.

[118] L. Xu, S. Hua, Z. Lin, Y. Liu, F. Ma, Y. Yan, X. Jin, X. Yang, and W. Zeng. "MotionBank: A Large-Scale Video Motion Benchmark with Disentangled Rule-Based Annotations". *arXiv preprint arXiv:2410.13790* (2024).

[119] W. Yin, R. Tu, H. Yin, D. Kragic, H. Kjellström, and M. Björkman. "Controllable Motion Synthesis and Reconstruction with Autoregressive Diffusion Models". *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2023, pages 1102–1108.

[120] T. Yoshikawa, V. Losing, and E. Demircan. "Machine Learning for Human Movement Understanding". *Advanced Robotics* 34.13 (2020), pages 828–844.

[121] F. E. Zajac. “Muscle and Tendon: Properties, Models, Scaling, and Application to Biomechanics and Motor Control”. *Critical Reviews in Biomedical Engineering* 17.4 (1989), pages 359–411.

[122] L. Zhang, H. Du, Z. Qin, Y. Zhao, and G. Yang. “Real-Time Optimized Inverse Kinematics of Redundant Robots under Inequality Constraints”. *Scientific Reports* 14.1 (2024), page 29754.

[123] M. Zhang, D. Jin, C. Gu, F. Hong, Z. Cai, J. Huang, C. Zhang, et al. “Large Motion Model for Unified Multi-Modal Motion Generation”. *Computer Vision – ECCV 2024*. Edited by A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol. Volume 15071. Lecture Notes in Computer Science. Cham: *Springer Nature Switzerland*, 2025, pages 397–421.

[124] J. Zhao, C. Wang, and B. Xie. “Human-like Motion Planning of Robotic Arms Based on Human Arm Motion Patterns”. *Robotica* 41.1 (2023), pages 259–276.

[125] J. Zou. “Multi-Scale Incremental Modeling for Enhanced Human Motion Prediction in Human-Robot Collaboration”. *arXiv preprint arXiv:2412.11632* (2024).