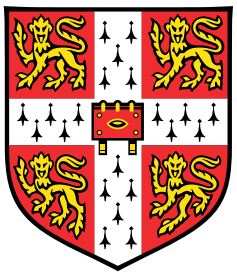
3D Human Motion Synthesis with Recurrent Gaussian Processes



Yeziwei Wang

Supervisor: Dr. Zhenwen Dai

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy

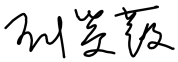
Clare College August 2018

I would like to dedicate this thesis to my loving parents.

Declaration

I, Yeziwei Wang of Clare college, being a candidate for the MPhil in Machine Learning, Speech and Language Technology, hereby declare that this report and the work described in it are my own work, unaided except as maybe specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

This dissertation contains 11,660 words excluding bibliography, figures, but including tables, footnotes, equations and apendices.



16/08/2018

Yeziwei Wang
August 2018

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Zhenwen Dai, for his guidance and countless support throughout the research and write-up. His insights have been exceptionally valuable. I would also like to thank Prof. Bill Byrne for his help during this project. Last but not least, I would like to acknoledge the support I recieved from friends, who made my journey more meaningful.

Abstract

3D human motion synthesis is the process of a model generating human motion sequences. Synthesised motions can be used in various applications, such as animation and video game production, people tracking, and human-machine interaction. 3D motions generated by motion capture usually require special equipments and actors performing movements. Thus, this process can be expensive and time-consuming. With the increasing popularity of machine learning methods in numerous applications, the utilisation of machine learning in motion generation becomes more common. The state-of-the-art synthesised motion is produced by Recurrent Neural Networks that are good at sequential modelling[7, 19, 15, 18]. Nonetheless, the deterministic model lacks the ability to incorporate uncertainty through the sequences. On the contrary, probabilistic models, such as Gaussian Processes, can deal with uncertainty. The base model used in the research is a Recurrent Gaussian Process that is capable of propagating uncertainty through a sequence[20].

The original RGP model[20] is modified into several variations and experimented on both toy data and motion capture data. Moreover, stochastic variational inference is used in implementation for efficient training. The latent RGP model variation with one hidden layer has shown excellent performance. Its capability to predict and generate multiple motions through one training further justifies the excellent representation power of RGPs. Even though satisfying results are achieved, further studies are needed to improve the flexibility of such models. Finally, potential future work are discussed at the end of the thesis.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The increasing popularity of 3D animation and video games witnessed a rising demand for diversified and realistic 3D human motion generation. Traditionally, human movements in animation are created by animators drawing each key frame of the motion. This requires careful observation and study of the human movements in video footage in order to draw realistic human motions. As technology advances, an automation method using motion capture became popular among animators. Motion capture requires for an actor to wear a specially designed suit, which can capture optical as well as mechanical features during the movement, and perform certain tasks. The recorded motions are then mapped to a 3D character displayed on a computer. However, this process can be tedious and expensive.

As machine learning gains reputation in various industries on different types of problems, a new type of human motion generation method occurred - the use of machine learning method to produce 3D human motions. 3D human motion synthesis is the process of a model generating human motion sequences based on a short window of initial movement. Other than computer graphics, this process can be valuable in a variety of domains including human-object interaction in robotics and driver maneuver anticipation in autonomous driving.

In this work, the focus is to model 3D human motions with Recurrent Gaussian Processes (RGP) and compare with different variations in the architecture. The original RGP from [20] is modified into a more scalable model using stochastic variational inference. The latent RGP model proposed in the dissertation shows satisfying performance, which is able to model multiple motions through one training. This model is also flexible to generate different motions, which means it does not require modifications in its structure to accommodate the training of different human motions. The data used in the modelling is the motion capture (Mocap) data from Carnegie Mellon University(CMU) graphics lab[1]. The synthesised

motions produced in this reasearch are publicly available at https://drive.google.com/open?id=1J38DbMuSAb_UEHUEcl9ea3GIo1qKYqF3.

1.1 Background

There are two types of models used in numerous machine learning tasks: deterministic models and probabilistic models. Deterministic sequence modelling, such as state space models and neural networks, are popular and successful among speech and language modelling. The state-of-the-art performances of human motion prediction and generation are achieved by deep Recurrent Neural Networks(RNN). The recurrent and deep structure are able to capture the complex human dynamics of spatial structure and temporal dependency shown in several variants of RNN[15, 19, 28, 10, 7]. However, the deep networks is difficult to train and require a large training dataset. Errors can easily be accumulated through the deep structure and have a significant impact on the output. To avoid accumulating errors some authors add noise to the input during training, which is to manually inject stochasticity into the model. On the other side, Human motions are embedded with uncertainty and variability as no two people have the exact same movements when performing one motion. Gaussian processes as probabilistic models are able to capture and propagate the inherent stochasticity of human motion. Given successful modelling, it is favourable to use recurrent structure in human motion modelling.

Therefore, RGP is a good fit for the task due to its recurrent structure and probabilistic feature. Simply, RGP is a Gaussian process with inherent recurrent structure making it an autoregressive model. This makes it different from traditional Gaussian processes in that the output of previous time step is used as input for current time step. The RGP model shows excellent performance in system identification tasks[20]. The paper mainly focuses on testing the model with system identification examples rather than motion sequences. Nonetheless, this thesis systematically studies the application of RGP on human motion modelling. One of the major differences between the two problems is the dimensionality difference, where human motion has a much higher dimension. Furthermore, the modelling of system identification also tries to learn the dynamics of the system and its relationship with the input control signals. However, human motions do not have a direct control signal, thus should not be simply treated the same as system identification. Despite the difference the model is able to generate acceptable predictions of human motions.

1.2 Motivations

The main motivation of this research is its potential applications in various fields. For example, both the film and video game industry will benefit from the successful scalable modelling of natural 3D human motions. The excessive cost and long hours of recording for motion capture can be replaced by a few hours of model training. In addition, this work has potential application in security, where computer vision tasks like people tracking is common and non-trivial. Moreover, more applications can be extended into medical industry, where motion monitoring can help predict and prevent the occurrence of stroke or cardiac arrest. These events are very time-critical, early detection can make a great difference in treatment. Last but not least, the model developed in this work is suitable for other sequence modelling. Potentially, it can also be used to generate music or speech sequences.

1.3 Structure of the Chapters

The remaining chapters are organised as follows. Chapter 2 focuses on reviewing recent literature. The literature review first covers the problem of 3D human motion generation and the models used for motion synthesis and prediction tasks. The second part of this chapter reviews literature about Gaussian processes, as well as some variations around them. Chapter 3 continues to introduce the theories involved in the research in detail. Interpretation of Mocap data is firstly introduced, followed by detailed explanation of Gaussian process and models based on Gaussian processes such as recurrent Gaussian processes and deep Gaussian processes. Sparse Gaussian process formulation and variational inference are both described in detail. Chapter 4 includes all the experiments, results and discussion on both toy data and Mocap data, Finally, potential directions for future work are discussed and suggested in chapter 5.

Chapter 2

Literature Review

Recent years of advancements in machine learning inspired interests in using machine learning methods to predict and generate 3D human motions, known as 3D human motion synthesis. Human motions can be represented in both 2 dimensions and 3 dimensions. 2D human motion is common in computer vision tasks such as recognising human poses from 2D images and people tracking in videos. 3D motions are commonly used in animations and video games. Many factors need to be taken into consideration when modelling human motions as human body is highly structured and sequences of motions propagate uncertainty through long time horizon.

Various models have occurred in literature modelling 3D human motions. One of the two major types of models is the parametric models, where the generation of motions is deterministic, for example dynamical models[7] and state space models[6]. Some recent models focus on deep learning structure, therefore models evolving around recurrent neural networks become popular. The other major type of models is non-parametric mainly evolving around Gaussian processes, as GPs can accommodate uncertainty within long-term motion sequence. Several variations aim to improve the scalability of the model, such as Gaussian process dynamical models[30] and recurrent Gaussian processes[20].

This chapter reviews some of the models proposed in recent literature. Since the base model for this project is a RGP, the literature review will also include Gaussian process and related algorithms.

2.1 3D Human Motions

There are limited literature that analyse the structure and details of human motions in Mocap dataset. However, [14] explores the features of joint movements of a hand performing natural tasks. As both the skeleton and the hand have the same representation, the analysis is transferable. The joints are represented using approximately 20 degrees of freedom (DOF). The paper shows that the dimensionality of the whole hand can be reduced to two major components using Principle Component Analysis(PCA), which is greatly influenced by the correlations between joints. For the fingers, more correlations are observed among joints closer to the palm. On the other hand, joints closer to the finger tip are more correlated within that finger. Similar observations can be found in human body joints.There are more correlations for joints closer to the hip and less for joints that are further away. Furthermore, the Mocap human skeletons also use degrees of freedom to represent each joint as a local representation with respect to its parent. Fully incorporate these correlations in the model is challenging, and some of the common models for human motion modelling are reviewed as follows.

2.1.1 State Space Models

State Space Models (SSMs) are commonly used in time series modelling, which contain two mappings: transition function and output function. Traditionally, they are both deterministic. As it is often required to propagate or predict uncertainty in many modelling scenarios, introducing uncertainty into SSMs becomes beneficial. Therefore, GP priors are introduced to both transition function and output function in SSMs forming GP-SSMs. By introducing GPs into SSMs, they become more suitable for modelling tasks such as human motion synthesis where uncertainty is non-trivial. [8] proposed a variational formulation with GP-SSMs, which provides an approximate posterior leading to fast probabilistic predictions. More recently, [6] proposed a probabilistic recurrent SSM(PR-SSM) variation, which offers efficient learning by combining gradient-based and sample-based inference.

A similar structure for modelling discrete-time non-linear system is non-linear autoregressive models with exogenous inputs(NARX), where the output depends both on the input and its own sequence history. The mapping in a NARX network can be any non-linear functions giving rise to different model variations. For example, the non-linear function can be a neural network resulting in a NARX recurrent neural network[18]. Placing a Gaussian process prior on the non-linear function gives Gaussian process NARX (GP-NARX) model.

2.1.2 Dynamical Models

In mathematics, a dynamical system is one that uses a function to describe the time dependency of a point in a geometrical space. The idea is used to form dynamical models which describe the dynamics of a model evolving through time like human motion modelling. [2] proposed using dynamical models to recognise human poses, which uses linear dynamical mappings to describe human motions over time. [24] further proposed using kernels to learn the parameters of the dynamical models. Due to the complexity and high dimensionality of human motions, switching linear dynamical models were proposed to generate smoother human motions by switching between different dynamical models that can better describe a specific motion[22].

The Gaussian Process Dynamical Model (GPDM), an extension from traditional dynamical model, has a latent space of 3 dimensions including a mapping that describes the transition of latent coordinates through time and a high dimensional observation space where human motion sequences lie[30]. GPDM can also be viewed as a type of SSM, where the non-linear mapping functions are linear combinations of basis functions where A and B are corresponding mapping parameters in Fig 2.1[30, 31].

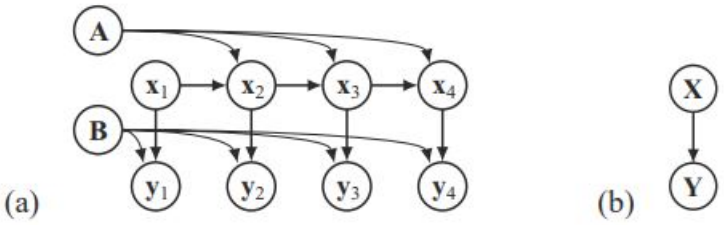


Fig. 2.1 Gaussian Process Dynamical Models. (a) A and B are the mapping parameters of the latent dynamics and observation mapping. (b) Summary of GPDM model, where the mapping parameters are marginalised over giving joint distribution of latent coordinates and observed poses[30, 31].

2.1.3 Recurrent Neural Networks

Many recent works on 3D human motion prediction and generation focus on deep neural networks. The state-of-the-art human motion modelling is generated by Recurrent Neural Networks(RNNs). Although RNNs present excellent performance in speech applications, some modifications are made to their original architecture in order to better suit human motion modelling tasks.

[7] proposed the Encoder-Recurrent-Decoder(ERD) to predict the probability of human motion of the next time instance. Within this network, the input is first fed to an encoder transforming it into a representation where learning of motion dynamics is easy. The recurrent layer takes the encoded representation as input and produce predictions which are then passed into a decoder to be transcribed into readable form. Similar structures are adopted by [3, 13]. [15] takes a step further adding higher level spatial-temporal structure into RNNs giving them the ability to interpret high dimensional structures in human motions. Looking at motion modelling from a different angle, [19] proposed to improve short-term human motion prediction using RNNs by modelling velocities of joints instead of their absolute angles.

Furthermore, a Dropout Auto-Encoder LSTM(DAE-LSTM) is proposed by [10] that consists of two components: firstly, a RNN is used to model the time dependency of the motion; secondly, a trained auto-encoder is used to recover spatial structures of human skeletons. The two components are trained separately with the latter trained by removing random joints in a skeleton and the former trained to predict future poses. Instead of looking at changes to the overall architecture, [28] modifies parts within the architecture. It introduces a temporal attention block to compute a history representation by giving different weights to historical skeletons. This history is than fed into a Modified Highway Unit(MHU) which selectively train the model based on activity-selected joints.

All above RNN-based models have one common limitation in spite of good results for motion modelling. They all lack the ability to cope with uncertainty within the motion sequences. This uncertainty can be introduced by different subjects or by the inherent stochasticity of human motion. Therefore, the combination of recurrent structure and GP becomes a reasonable proposal for the task.

2.2 Gaussian Processes

Gaussian processes are widely used in geostatistics field where it is known as kriging[25]. Gradually, GP prediction is used in general regression. Due to the increasing developments in machine learning in the past years, GP was introduced into into machine learning by [32], which also describes the optimisation of parameters in covariance function. Since then GPs are commonly used for various regression problems, as well as for classification. In recent years, variations of GPs and emerging approximation methods are continuing to enrich the GP community.

2.2.1 Gaussian Process Latent Variable Models

GP latent variable models (GPLVM) are proposed by [17], which is mainly used for dimension reduction in high dimensional data visualisation. GPLVM is essentially a generalisation of the underlying probabilistic model for PCA. Manipulating covariance function can turn GPLVM into PCA simply by constraining the covariance function to linear mapping. As GPLVM allows non-linear covariance mapping and uncertainty in the latent space, it shows more flexibility and robustness in dimension reduction tasks.

2.2.2 Recurrent Gaussian Processes

Although GPLVMs perform well with visualisation of high dimensional data, they lack the ability to deal with sequences, such as human motions. This is where RGP suffice by combining recurrent structure with GPs[20]. The paper develops a general non-parametric model with recurrent GP priors and recurrent variational Bayes framework to represent sequential data. The model is able to learn from a small dataset compared to parametric models such as neural networks. The general RGP structure proposed by [20] is shown in 2.2.

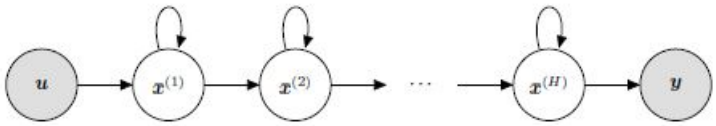
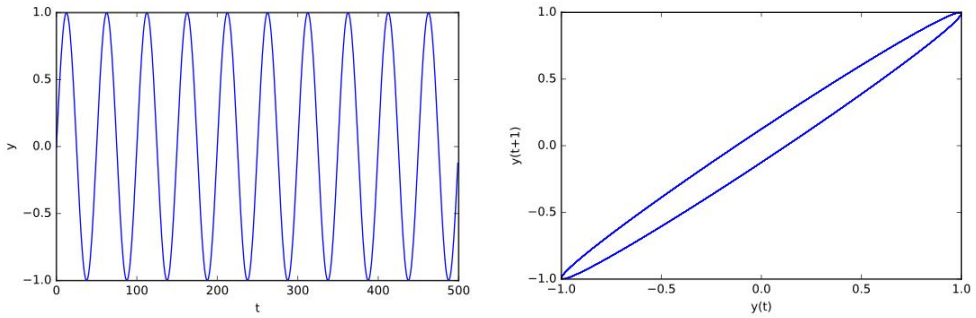


Fig. 2.2 Recurrent Gaussian Process[20]

2.2.3 Gaussian Process Autoregressive Regression Model

In a recent literature[16], an autoregressive GP, named Gaussian Process Autoregressive Regression model (GPAR), is presented as a multi-output regression model. Despite the outstanding tractability and interpretability of GPs, it is based on the assumption that the mapping between input and output is one-to-one, meaning there is a single output at each input location. However, it is typical for functions to have multiple outputs. For example, a sine wave in the autoregressive setting shown in Fig 2.3. Along time axis, a sinusoidal waveform has one value at each time step. However, when the signal becomes autoregressive, that is $y(t+1) = f(y(t))$, there are two outputs at each input position. GPARs propose to deal with this type of mappings as well as capturing the dependencies between these outputs.



(a) (b)

Fig. 2.3 (a) Sine wave along time. (b) Sine wave along its previous value.

2.2.4 Deep Gaussian Processes

As deep learning becomes increasingly popular, this structure is transferred into GPs giving Deep Gaussian processes (DGPs). [5] mentions that deep learning structure are mainly associated with Restricted Boltzmann Machine (RBM) based models. Given that the output of RBMs only depends on a linear combination of the inputs, GPs show better representation power because the likelihood is over a continuous variable space and the output is a non-linear mapping of the input. The main contributions of the aforementioned paper is realising truly deep hierarchies using variational approximations and its applicability on sparse data. [20] also used a deep structure in its modelling, where all the latent layers are recurrent, shown in Fig 2.2. However, the RGP deep structure vary from the DGP in [5] where the former only requires passing on values from layer to layer and the latter requires a GP mapping between the layers.

2.2.5 Learning Tricks

Sparse Gaussian Processes

Original full GP regression models has a kernel that increases significantly in size as the amount of training data increases. This no doubt becomes a non-trivial problem in GP regression learning. A sparse GP is described in [26] used to approximate the kernel with some chosen input points, called pseudo-inputs. [29] later proposed variational learning method to learn kernel hyper parameters and inducing inputs simultaneously by maximizing the lower bound of log marginal likelihood.

Variational Inference

In models with hidden variables, the posterior is not tractable in a lot of the cases. Variational inference is one of the most prominent approximation method appealed to. As data sets become increasingly large, traditional variational inference become extremely slow. This is because to give an estimate for the value of kernel parameters, all input points must be evaluated, thus more input points means more computing time. Stochastic variational inference is proposed in [12] with specific application in latent Dirichlet allocation and the hierarchical Dirichlet process topic model. [11] introduced statistical variational inference on Gaussian processes and presented toy examples along with real life examples. With the use of stochastic variational inference, hyper-parameters can be learned by mini-batch training as estimates do not require complete training data. This makes the use of variational inference scalable.

Both sparse GP and statistical variational inference are used in modelling 3D human motions within this research. The details behind these two methods are further discussed in the theory section together with other relevant models and techniques. All above reviewed literature contribute to the multiple variations of models predicting human movements.

Chapter 3

Theory

This Chapter begins with an introduction of Mocap dataset, its structure and how it is used within the modelling. Then, the theory behind Gaussian processes is introduced more formally in Chapter 3.2 from the basic Gaussian distribution. In the following sections, further variations based on GPs are explained, such as Deep Gaussian Processes(DGPs) and Recurrent Gaussian Processes(RGPs). Sparse GP is also introduced as an approximation method to reduce the computation complexity of a full GP, followed by variational inference and its application in the RGP model. At the end of the chapter stochastic variational inference with mini-batch training is briefly introduced as a training method for efficient hyper-parameter learning.

3.1 Motion Capture Dataset

Within Mocap dataset, human motions are represented by skeletons, whose joints are described by degrees of freedom (DoF). DoF is a local representation of each joint and poses constraints on the movement of the joint. This is natural as human joints have limited range of movement. Human motions in Mocap dataset have pair-wise representations, such as .asf/.amc files. The former describes the hierarchical structure of a skeleton and its joints, whereas the latter stores the movement information of the skeleton[1]. Namely, the .amc file has values of each joint of the skeleton at each time step, thus describing the motion of the skeleton.

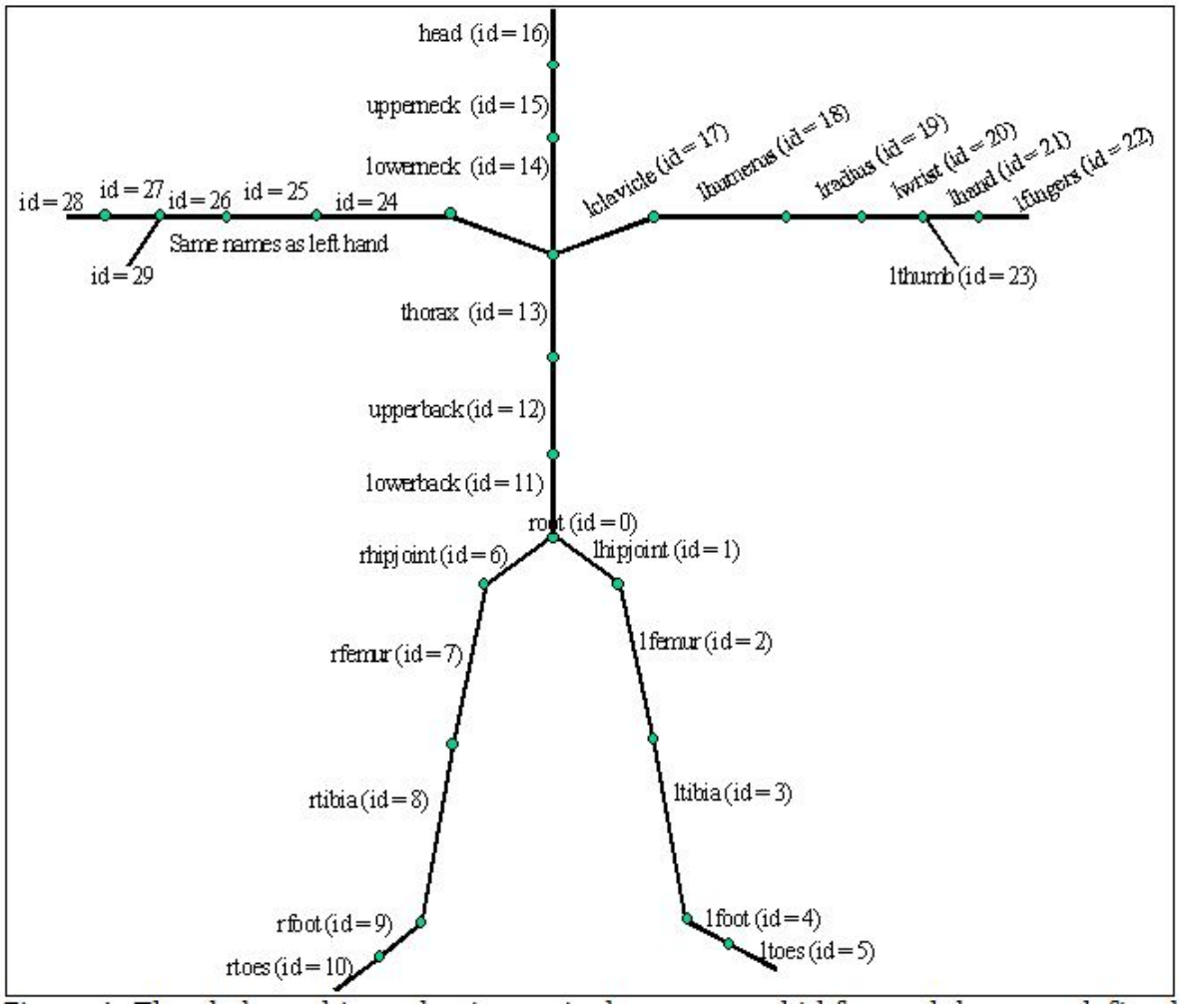


Fig. 3.1 Hierarchical skeleton representation from .asf file [1]

The joints and skeleton structure described in the acclaim skeleton files (.asf) can be represented in Fig 3.1. The *root* joint contains information in relation to the global coordinate using the axis-angle representation that contains 6 values: TX, TY, TZ, AX, AY, AZ. The first three values are the 3D global coordinates and the latter three are angles encoding the orientation of the torso. Other than *root*, all other joints use local angle representation, degree of freedom, with respect to their parent, which are constrained by rotation ranges. As shown in Fig 3.1, where each joint has a unique ID, the parent of a joint is its previous joint according to the ID sequence. For example, *upperback*(ID=12) is the parent of *thorax*(ID=13). There are 29 joints in total, where each joint is represented by a vector of different dimensions. For example, *root*(ID=0) is a joint of 6 dimensions and head(ID=16) is a joint of 3 dimensions. These values of 62 dimensions in total are modified in the .amc file across all frames to describe 3D movements of the skeleton across time steps.

The 3D motion data used in the modelling are the locations and angles extracted directly from the .amc file. These local representations are used to train and test various model architectures of RGPs.

3.2 Gaussian Processes

A Gaussian Process (GP) can be seen as a distribution over functions[25]. GPs can be used in various problems including regression and classification. Unlike most methods that describe a function as a deterministic mapping between an input and an output, GPs provide an probabilistic view of describing a function. The representation and understanding of GPs can be expanded from the basic Gaussian distributions.

3.2.1 Gaussian Distribution

A Gaussian distribution is the most common distribution in statistics and machine learning due to its intuitiveness and mathematical tractability. A variable x that is Gaussian distributed is usually expressed as

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \tag{3.1}$$

where $\mu = \mathbb{E}[x]$ is the mean of the distribution and $\sigma^2 = var(x)$ is the variance of the distribution. For the Gaussian distributed variable x, the most likely value is the mean of the distribution. The further away from the the mean, the less likely is x going to take that value. A Gaussian distribution with zero mean and 0.5 variance is shown in Fig 3.2 and the area underneath the curve sums up to 1.

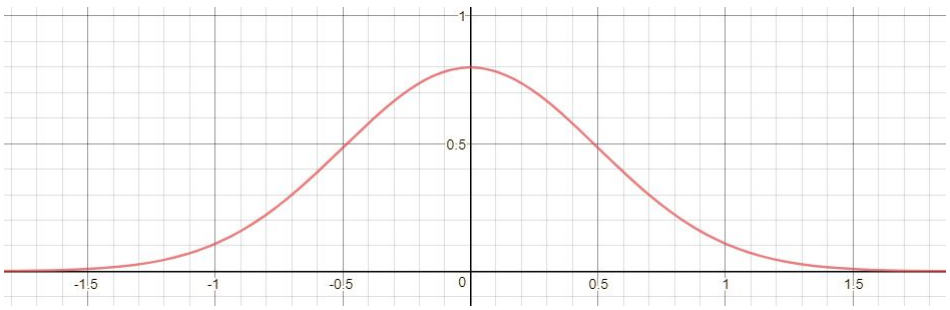


Fig. 3.2 Gaussian Distribution, $\mu = 0$, $\sigma^2 = 0.5$

3.2.2 Multivariate Gaussian Distribution

When the random variable of a Gaussian distribution is a vector with D elements, it is a multivariate Gaussian distribution that has D variables. The marginalisation of each variable is a univariate Gaussian distribution as aforementioned. Therefore, each value in the mean vector corresponds to the mean of each variable and the variance of the multivariate distribution is a $D \times D$ covariance matrix as follows:

$$\mu = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_D \end{bmatrix}, \Sigma = \begin{bmatrix} k_{1,1} & \dots & k_{1,D} \\ \vdots & \ddots & \vdots \\ k_{D,1} & \dots & k_{D,D} \end{bmatrix} \tag{3.2}$$

Each element of covariance matrix describes the covariance between two variables. For example, $k_{i,j}$ is the covariance between variable x_i and x_j. The multivariate Gaussian distribution can then be expressed using the mean vector and covariance matrix:

$$\mathcal{N}(X|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} e^{-\frac{1}{2\sigma^2}(X-\mu)^\top \Sigma^{-1}(X-\mu)} \tag{3.3}$$

The shape and properties of multivariate Gaussian distributions are determined by the mean vector and covariance matrix. The normalization constant from Equation (3.3), $\frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}}$, is chosen in such way that the area underneath the distribution sums up to 1.

$$\frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} exp\Big(-\frac{1}{2\sigma^2}(X-\mu)^\top \Sigma^{-1}(X-\mu)\Big) = 1 \tag{3.4}$$

An example of multivariate Gaussian distributions with 2 variables, x_a, x_b, is shown in Fig 3.3. The bivariate Gaussian distribution has zero mean vector and diagonal covariance matrix.

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

The covariance matrix defines that the two variables are independent from each other, as the covariance between two variables equals to 0. Thus, the distribution has a perfectly even bell shape centred at coordinate origin.

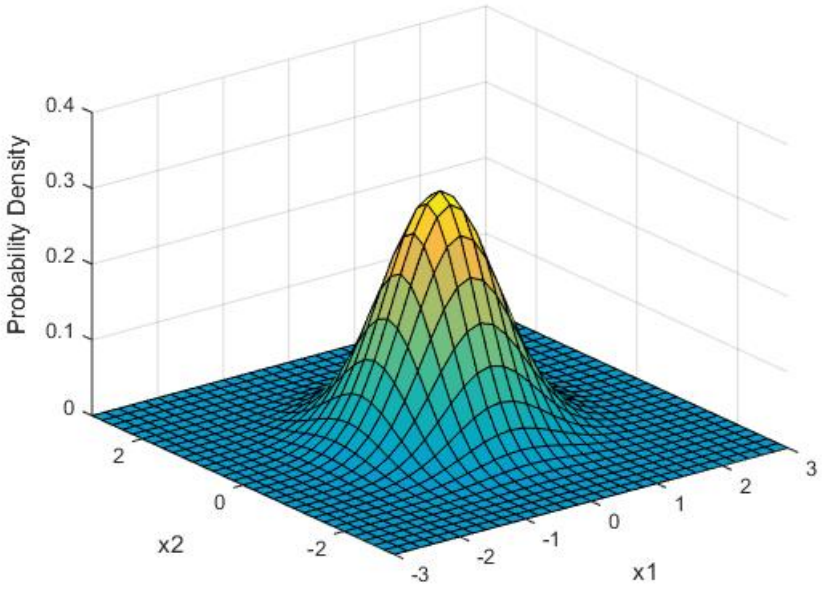


Fig. 3.3 A Bivariate Gaussian distribution

A Gaussian distribution has nice properties that makes it a popular choice in many statistics and machine learning problems. For example, marginalization and conditional distribution of a Gaussian are still Gaussian distributed. Given random means and variances for variable x_a and x_b, the joint bivariate distribution has the following decomposition

$$\mu = \begin{bmatrix} \mu_a \\ \mu_b \end{bmatrix}, \Sigma = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ba} & K_{bb} \end{bmatrix} \tag{3.5}$$

Therefore, the marginals are

$$\begin{aligned} p(x_a) &= \int p(x_a, x_b) dx_b = \mathcal{N}(\mu_a, K_{aa}) \\ p(x_b) &= \int p(x_a, x_b) dx_a = \mathcal{N}(\mu_b, K_{bb}) \end{aligned} \tag{3.6}$$

Also, the conditional densities can be expressed as

$$\begin{aligned} x_a | x_b &\sim \mathcal{N}(\mu_a + K_{ab} K_{bb}^{-1}(x_b - \mu_b), K_{aa} - K_{ab} K_{bb}^{-1} K_{ba}) \\ x_b | x_a &\sim \mathcal{N}(\mu_b + K_{ba} K_{aa}^{-1}(x_a - \mu_a), K_{bb} - K_{ba} K_{aa}^{-1} K_{ab}) \end{aligned} \tag{3.7}$$

Take the example in Fig 3.3, if the value of x_a is known, then the values of x_b can be inferred based on the conditional distribution of x_b given x_a. In a multivariate distribution, the random

variables are discrete. However, in real world modelling, the distribution is usually over a continuous space where the variables are no longer discrete. Gaussian processes is used in this scenario, where the variable space is continuous and infinite.

3.2.3 Gaussian Processes

Compared to multivariate Gaussian distributions, the mean and covariance function of a GP replace the mean vector and covariance matrix of a Gaussian distribution. Hence, a sample from a GP is a function, whereas a sample from a multi-variate Gaussian distribution is a n-dimensional vector. Conveniently, GPs can be seen as the generalization of multivariate Gaussian distribution. The formal definition of GPs from [25] is as follows:

Definition 1. *A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

A GP, $\mathcal{GP}(x)$, can be fully specified by a mean function and covariance function, where its mean function $m(x)$ and covariance function $k(x,x')$ can be defined as

$$f(x) \sim \mathcal{GP}\big(m(x), k(x,x')\big) \tag{3.8}$$

$$m(x) = \mathbb{E}\big[\mathcal{GP}(x)\big], \tag{3.9}$$

$$k(x,x') = \mathbb{E}\big[(\mathcal{GP}(x) - m(x))(\mathcal{GP}(x') - m(x'))\big], \tag{3.10}$$

From the expression in Equation (3.9), the random variables in GP represent the value of a function $f(x)$ at location x. Although it is common to define GPs over time, i.e. the input variable x represents time, the use of GPs in this thesis adopts the autoregressive form. This will be further explained in section 3.4. Moreover, the covariance function specifies the covariance between a pair of random variables,

$$cov(f(x_1), f(x_2)) = k(x_1, x_2) \tag{3.11}$$

The covariance function can have a set of hyper-parameters to determine the mapping from arbitrary inputs to the covariance domain. It is common to assign the mean of a GP to zero as an uninformative prior. This is because usually we have little knowledge of the prior of the distribution we are trying to solve. Zero mean also gives a simpler form of expression, which leaves the distribution itself to be solely described by its covariance function. The

hyper-parameters in the covariance functions, therefore, become essential for modelling. More information about covariance functions will be given in section 3.2.3.

In real life modellings, despite the infinite representation power of GPs, only finite number of data points are available in the input space. Therefore, the model is specified in a form similar to a multivariate Gaussian distribution, but on a much larger scale. This utilises the marginalisation property of Gaussian distributions, assuming the rest of the points in the input space are marginalized out. This gives a finite representation of GPs, assuming zero mean

$$p(f|x) = \mathcal{N}(f|0, K_{ff}) = \frac{1}{\sqrt{(2\pi)^N K_{ff}}} exp\big(-\frac{1}{2} f^\top K_{ff}^{-1} f\big) \tag{3.12}$$

The covariance matrix $K_{ff} = k_f(x,x)$ is calculated over all N training inputs, which makes K_{ff} a $N \times N$ matrix.[1] $f(x)$ is an observation based on input x and the observation follows a Gaussian distribution. In the ideal case, observations can be noise free, which have the density described in Equation (3.12). However, in realistic applications, it is typically assumed that observations are noisy. Both cases are further discussed in the following sections.

Noise-free observations

If the observation of a GP is noise-free, the observation is completely determined by the random function sampled from a $\mathcal{GP} : f = f(x)$[21]. The prediction joint distribution is decomposed into

$$\begin{bmatrix} f \\ f_* \end{bmatrix} = \begin{bmatrix} 0, \begin{bmatrix} K & K_* \\ K_*^\top & K_{**} \end{bmatrix} \end{bmatrix} \tag{3.13}$$

where the f is the training observations and f_* is the prediction on n_* testing inputs. Naturally, μ_* is the mean of the prediction and $K_{**} = k(x*, x*)$ is the covariance matrix computed from all testing points. Similarly, K_* denotes the $n \times n_*$ covariance matrix evaluated between all training and testing data pairs. Thus, the predictive posterior distribution conditioned on all observations can be written as

$$p(f_*|X_*, X, f) = \mathcal{N}(f_*|\mu_*, \Sigma_*) \tag{3.14}$$

[1] Here, for simplicity reasons, the expression ignores the dependency of covariance function on kernel hyper-parameters.

where

$$\mu_* = K_*^\top K^{-1} f \tag{3.15}$$
$$\Sigma_* = K_{**} - K_*^\top K^{-1} K_* \tag{3.16}$$

Noisy observations

Realistically, it is more common to consider noise in the observations. The noisy observations are often written as $y = f(x) + \varepsilon$, where the noise ε has zero mean and diagonal covariance matrix. For example, for one of the output dimension, $\varepsilon_i \sim \mathcal{N}(0, \sigma_y^2)$. The covariance matrix of the noisy observations given some data points is written as

$$K_Y = cov(Y|X) = K + \sigma_y^2 I_N \tag{3.17}$$

The prediction joint distribution can be expressed as

$$\begin{bmatrix} Y \\ f_* \end{bmatrix} = \begin{bmatrix} 0, \begin{bmatrix} K_Y & K_* \\ K_*^\top & K_{**} \end{bmatrix} \end{bmatrix} \tag{3.18}$$

The posterior predictive distribution density function given all noisy observations is

$$p(f_*|X_*, X, Y) = \mathcal{N}(f_*|\mu_*, \Sigma_*) \tag{3.19}$$

where

$$\mu_* = K_*^\top K_Y^{-1} Y \tag{3.20}$$
$$\Sigma_* = K_{**} - K_*^\top K_Y^{-1} K_* \tag{3.21}$$

The kernel parameters are the hyper-parameters of a GP. During training the optimal hyper-parameters are often estimated by maximizing marginal likelihood, which can easily be obtained according to Gaussian distribution's marginalization property:

$$p(Y|X) = \int p(Y|f, X) p(f|X) df \tag{3.22}$$

As $p(f|X)$ and $p(Y|f)$ are known as

$$p(f|X) = \mathcal{N}(f|0, K), \quad p(Y|f) = \prod_i \mathcal{N}(y_i|f_i, \sigma_y^2) \tag{3.23}$$

the log-marginal likelihood can be written as

$$\begin{aligned}\log p(Y|X) &= \log \mathcal{N}(Y|0,K_Y) \\ &= -\frac{1}{2}Y^\top K_Y^{-1}Y - \frac{1}{2}\log|K_Y| - \frac{N}{2}\log(2\pi) \\ &= -\frac{1}{2}Y^\top (K+\sigma_y^2 I)^{-1}Y - \frac{1}{2}\log|K+\sigma_y^2 I| - \frac{N}{2}\log(2\pi)\end{aligned} \tag{3.24}$$

Gradient descent is usually used to optimize the log-marginal likelihood. As latent variables are introduced into a GP-based model, log-marginal likelihood can no longer be solved easily. Approximation methods need to be considered to estimate posterior distribution of latent variables. Variational inference is commonly applied during model training. Other GP-based models and variational inference will be further explained later in this chapter.

Covariance Functions

A GP with zero mean is solely determined by its covariance function. One commonly used covariance functions is squared-exponential kernel, also called Radius Basis Function(RBF) kernel. One dimensional RBF can be expressed as

$$k_y(x_i,x_j) = \sigma_f^2 exp(-\frac{1}{2l^2}(x_i - x_j)^2) \tag{3.25}$$

where l is the length-scale parameter and σ_f^2 is the signal variance. These are the hyper-parameters mentioned in the previous sections. The influence of the hyper-parameters are shown in the following plots. Fig 3.4 shows a GP prior with $(\sigma_f^2, l, \sigma_n^2) = (1, 0.1, 0.1)$ fitting data points given by x, where σ_n^2 is the noise variance.

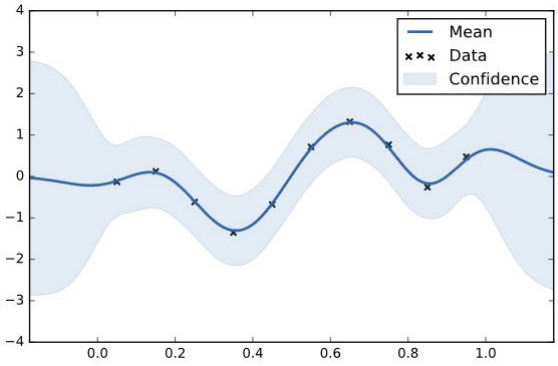
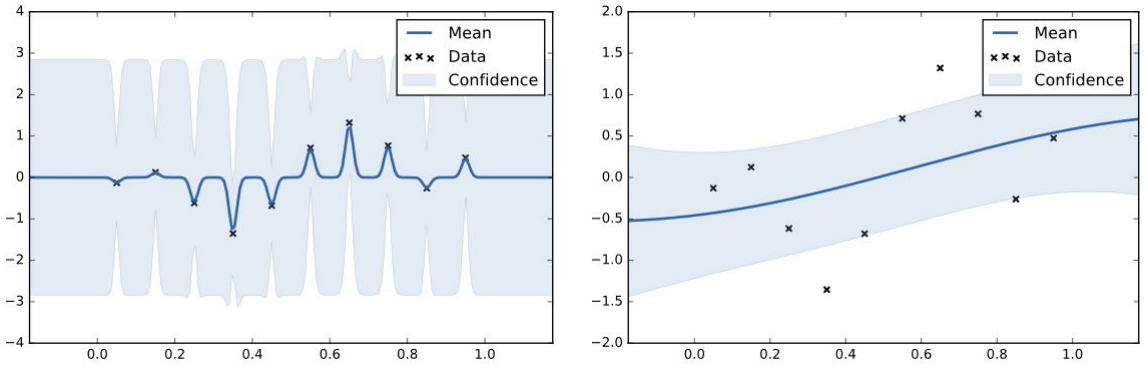


Fig. 3.4 A one dimension Gaussian process with following parameters: length scale=0.1, signal variance=1, noise variance=0.1

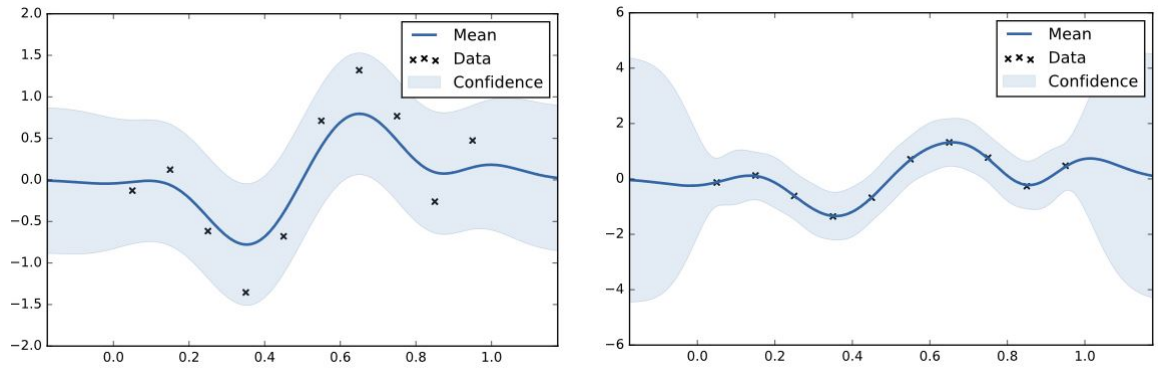
In general, length scale determines how relevant the data points are to each other. Namely, how far apart should two input points be until they become uncorrelated. As shown in Fig 3.5, the length scale varies while the other two parameters stay the same. When length scale is large, most input points are highly correlated meaning the curve is smooth. Whereas small length scale makes the curve very wiggly, because input points are mostly independent from each other.



(a) (b)

Fig. 3.5 One dimension GP with varying length scale. (a) has length scale of 0.01. (b) has length scale of 1.

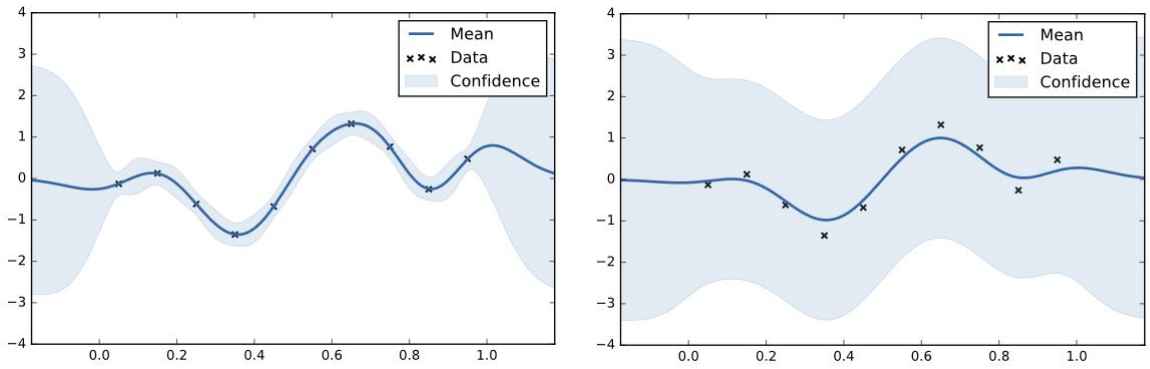
When the signal variance is small, a larger noise variance is needed to account for all the data points. On the contrary, large signal variance means smaller noise variance is needed to fit the data. This is shown in Fig 3.6 with fixed length scale and noise variance.



(a) (b)

Fig. 3.6 One dimension GP with varying signal variance. (a) has signal variance of 0.1. (b) has noise variance of 5.

The effect of noise variance is shown in Fig 3.7 with fixed length scale and signal variance as baseline shown in Fig 3.4. When noise variance is high, there is more uncertainty in curve fitting around each observation. On the contrary when noise variance is low, the curve has more certainty around the given data points.



(a) (b)

Fig. 3.7 One dimension GP with varying noise variance. (a) has noise variance of 0.01. (b) has noise variance of 1.

Assume the x_i, x_j in Equation (3.25) are both D-dimensional input vectors from input space, the RBF kernel can also be expanded into the following form

$$k_y(x_i, x_j) = \sigma_f^2 exp\left(-\frac{1}{2l^2} \sum_{k=1}^{D} (x_{i,k} - x_{j,k})^2 \right) \tag{3.26}$$

In this expression, all input dimensions has the same length scale l. Realistically, all dimensions may have different length scale to incorporate complex high-dimensional data. Thus, the scaler length scale becomes a vector of D dimension: $l = (l_1, l_2, \ldots, l_D)$. A new notation is introduced for convenience, where $\omega_k = 1/l_k^2$ denotes a new parameter as 'weight', resulting in the new form of RBF covariance function[4].

$$k_{y(ARD)}(x_i, x_j) = \sigma_{ARD}^2 exp\left(-\frac{1}{2} \sum_{k=1}^{D} \omega_k (x_{i,k} - x_{j,k})^2 \right) \tag{3.27}$$

This form introduces automatic relevance determination (ARD)[25] into kernel function. The weight ω_k is the inverse of length scale determining how relevant the inputs are along dimension k. If the weight is small, indicating a large length scale, the covariance will be small making the input more independent. When the weight is small enough, this effectively removes that input from inference. As the name suggested, by learning length scale of each

dimension the kernel has the ability to automatically determine the relevance of an input. This form of RBF kernel is used to describe the GPs during RGP modelling.

3.3 Sparse Gaussian Processes

For a GP regression problem with N inputs, its time complexity is $\mathcal{O}(N^3)$ during training and $\mathcal{O}(N^2)$ during prediction due to inverse of covariance matrix. As training dataset becomes large, training a GP can be slow due to high computation costs. Therefore, in recent literature a method of using few points $M \ll N$ to represent the full GP, known as inducing points, is widely used. The use of inducing points reduces the size of covariance matrix from $N \times N$ to $M \times M$. As a result, the computation cost during training and prediction is reduced to $\mathcal{O}(NM^2)$ and $\mathcal{O}(M^2)$[27, 9].

The reason sparse GP works is because not all points are equally informative for recovering a function. For example, a function that looks like the one in Fig 3.8. Only a few points are needed to capture the behaviour at the flat region where x is greater than 5, thus more points will not improve the regression posterior but increase the computational cost. A more reasonable approach is to use more points within the active region, where x is less than 5, and fewer points elsewhere. This is key idea behind sparse GP.

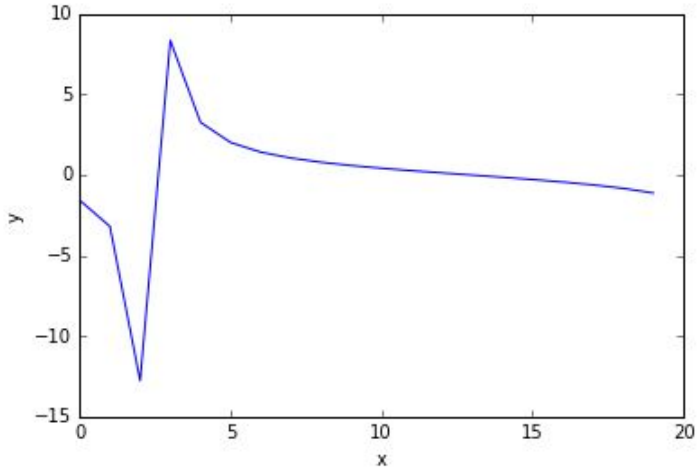


Fig. 3.8 Random Function

Some of the notations to describe sparse GP are introduced here to explain its formulation. Let's introduce M inducing points with Z denoting their locations. Let $U = [u_1, ...u_m]^\top$ denote the output values at the inducing points. The covariance matrix over the M inducing

points is written as K_{mm}. Similarly, K_{nm} is the covariance matrix between the input X and inducing points Z and K_{*m} is the covariance matrix between prediction points X_* and inducing points. In the predictive distribution, the GP posterior can then be approximated by using inducing points instead of the complete training data.

$$p(f_*|X_*,Y,X,Z) \approx \int \mathcal{N}(f_*|\mu_m,\Sigma_m,X_*,Z)p(U|Z,Y,X)dU \tag{3.28}$$

where

$$\begin{aligned}\mu_m &= K_{*m}K_{mm}^{-1}U \\ \Sigma_m &= K_{**} - K_{*m}K_{mm}^{-1}K_{m*}\end{aligned}$$

The inducing points Z can be treated as variational parameters in the above expression, where a variational lower bound is used instead of the true marginal likelihood. Then Z values can be optimised by maximising the lower bound. More details about variational inference with sparse GP will be included in section 3.6.

Several sparse approximation methods described in the literature are with different focus. A unifying view is provided in [23]. Different approximation methods vary by introducing different additional assumptions regarding the two approximation conditional probabilities. In this dissertation, details of these approximation variations won't be further discussed.

3.4 Recurrent Gaussian Processes

A graphical representation of a general RGP[20] is shown in Fig 2.2. Multiple hidden layers $x_{1:H}$ carry recurrent structures, where the mappings are sampled from GPs. Input layer, u, and output layer, y, are both observed, where input layer passes on information to hidden layers and output layer generates sequences based on the recurrent structure within the hidden layers. This is how a RGP is defined in [20]. A RGP with one hidden layer can be written as

$$\begin{aligned}x_i &= f(\bar{x}_{i-1},\bar{u}_{i-1}) + \varepsilon_i^x \\ y_i &= g(\bar{x}_i) + \varepsilon_i^y \\ \bar{x}_i &= [x_i,...x_{i-L+1}]^\top \\ \bar{u}_i &= [u_i,...u_{i-L_u+1}]^\top\end{aligned} \tag{3.29}$$

where L is the window size of the history in hidden layer, and L_u is the window size of the input signal. The mappings are samples from two different GPs, where

$$f \sim \mathcal{GP}(0, K_f) \tag{3.30}$$

$$g \sim \mathcal{GP}(0, K_g) \tag{3.31}$$

Both of the GPs can be represented using above mentioned sparse GPs. For 3D human motion modelling task, the models used are based on RGP architecture with some modifications. The details about the models will be further explained in the next Chapter. The detailed derivation of the RGP used can be found in the Appendix.

3.5 Deep Gaussian Processes

A Deep Gaussian Process (DGP) can be seen as a deep network with GPs as the mapping between layers. Overall, the DGP model is no longer a GP, but something more sophisticated. The depth is used to describe the number of layers in the DGP like neural networks. The variables in hidden layers are going to be treated as latent variables. The stacking structure of a DGP is shown in Fig 3.9. The GPs that correspond to different layers are different, hence a unique mapping is sampled for each layer. The output of a GP from each layer serves as the input of the GP of the next layer.

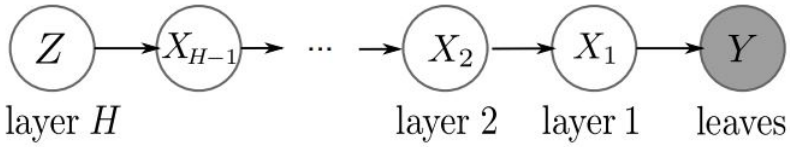


Fig. 3.9 A deep Gaussian process with two hidden layers [5]

In a more general case, for a DGP that has H layers the output Y can be expressed recursively as

$$Y = f_{1:H} + \varepsilon \tag{3.32}$$

$$Y = f_1(f_2((f_H(z))...)) + \varepsilon \tag{3.33}$$

where ε is the observation noise at the output. [4] also proposed a probabilistic definition for DGPs. The joint distribution of latent layers and the output is

$$p(y, h_{1:H}) = p(y|h_1)p(h_1|h_2)...p(h_{H-1}|h_H)p(h_H) \tag{3.34}$$

where $p(h_H)$ is the prior for the H_{th} hidden layer. The conditional probability of the intermediate layer is

$$p(h_{l-1}|h_l) = \int p(h_{l-1}|f_{l-1})p(f_{l-1}|h_l)df_{l-1} \tag{3.35}$$

The two terms in Equation (3.35) can be expressed using Equations in (3.23). Due to the intractability of the joint representation, approximation methods are often applied during inference. The details of variational inference will be discussed in the next section.

3.6 Variational Inference

Variational inference is used to approximate the intractable posterior. Assume that $x = x_{1:n}$ are observations and $z = z_{1:m}$ are hidden variables, the posterior distribution of the latent variables is

$$p(z|x) = \frac{p(z,x)}{\int_z p(z,x)} = \frac{p(z,x)}{p(x)} \tag{3.36}$$

A variational distribution, $q(z_{1:m}|\nu)$, determined by variational parameters is chosen to approximate the distribution over latent variables. The posterior is approximated using q with fitted parameters.

Kullback-Leibler (KL) divergence measures the distance between two distributions.

$$KL(q||p) = \mathbb{E}_q\big[\log \frac{q(z)}{p(z|x)}\big] \tag{3.37}$$

where $p(z|x)$ is the true posterior distribution and $q(z)$ is its variational approximation. In order to find a variational distribution that is closest to the original distribution, the KL divergence between the two distributions should be minimized. However, this cannot be achieved by minimizing KL divergence directly. Therefore, an alternative is used to maximize the evidence lower bound (ELBO) of observation sequence. Hence, minimising the KL divergence between the two distributions.

According to Jensen's inequality, the lower bound of the log marginal likelihood can be found,

$$\begin{aligned}\log p(x) &= \log \int p(x,z)dz \\ &= \log \int p(x,z)\frac{q(z)}{q(z)}dz \\ &= \log \mathbb{E}_q\big[\frac{p(x,z)}{q(z)}\big] \\ &\geq \mathbb{E}_q\big[\log p(x,z)\big] - \mathbb{E}_q\big[\log q(z)\big] = ELBO\end{aligned} \tag{3.38}$$

Considering the relationship between KL divergence and ELBO,

$$\begin{aligned}KL(q(z)||p(z|x)) &= \mathbb{E}_q\big[\log \frac{q(z)}{p(z|x)}\big] \\ &= \mathbb{E}_q\big[\log \frac{q(z)p(x)}{p(x,z)}\big] \\ &= \mathbb{E}_q[\log q(z)] - \mathbb{E}_q[\log p(x,z)] + \mathbb{E}_q[\log p(x)] \\ &= -ELBO + \log p(x)\end{aligned} \tag{3.39}$$

Thus, minimizing the KL divergence is equivalent to maximizing the ELBO.

There are many methods to approximate the variational distribution with respect to variance parameters. In [20], mean field approximation is used. In mean field variational inference, the variational distribution factorizes as

$$q(z_1,...,z_m) = \prod_{j=1}^{m} q(z_j) \tag{3.40}$$

The detailed derivation for RGP is included in the Appendix.

3.6.1 Variational Inference with Sparse Gaussian Process

Sparse GP is often used to represent a GP and variational inference is a common approximation method for GP inference. Thus, variational inference with sparse GP is introduced in detail in this section. The following notations are used for arriving at the objective function for training a sparse GP with variational inference. X denotes the input, Y denotes outputs corresponding to input positions, f denotes the mapping function sampled from GP, and U denotes the output of inducing points Z. Find the variational lower bound of the log marginal

likelihood after introducing inducing points and factorisation:

$$\begin{aligned}\log p(Y|X) &= \log \int p(Y|f)p(f|X,U)p(U)d(U,f) \\ &\geq \int p(f|X,U)q(U)\log\frac{p(Y|f)p(U)}{q(U)}d(U,f) \\ &= \mathbb{E}_q\big[\int p(f|X,U)\log p(Y|f)df + \log p(U) - \log q(U)\big]\end{aligned} \tag{3.41}$$

For simplicity reasons, the dependency of Z is dropped in $p(U|Z)$, giving $p(U)$. Each term in Equation (3.41) can be expanded as follows:

$$\begin{aligned}\int p(f|X,U)\log p(Y|f)df &= \int p(f|X,U)\sum_{i=1}^{n}\log p(Y_i|f_i)df \\ &= \sum_{i=1}^{n}\int p(f_1,\ldots,f_n|X,U)\log p(Y_i|f_i)d(f_1,\ldots,f_n) \\ &= \sum_{i=1}^{n}\int \log p(Y_i|f_i)\Big(\int p(f_{1:n}|X,U)d(f_{-i})\Big)df_i \\ &= \sum_{i=1}^{n}\int p(f_i|X_i,U)\log p(Y_i|f_i)df_i\end{aligned} \tag{3.42}$$

where f_{-i} refers to $f_1,\ldots,f_{i-1},f_{i+1},\ldots,f_n$.

$$\begin{aligned}p(Y_i|f_i) &= \mathcal{N}(Y_i|f_i,\beta^{-1}I) \\ &= (2\pi\beta^{-1})^{-d/2}exp\big(-\frac{\beta}{2}(Y_i-f_i)(Y_i-f_i)^\top\big)\end{aligned} \tag{3.43}$$

$$\begin{aligned}\int p(f_i|X_i,U)\log p(Y_i|f_i)df_i &= \mathbb{E}_{p(f_i|X_i,U)}\big[-\frac{d}{2}\log(2\pi\beta^{-1}) - \frac{\beta}{2}(Y_iY_i^\top - 2f_iY_i^\top + f_if_i^\top)\big] \\ &= -\frac{d}{2}\log(2\pi\beta^{-1}) - \frac{\beta}{2}Y_iY_i^\top + \beta\mathbb{E}_{p(f_i)}[f_iY_i^\top] - \frac{\beta}{2}\mathbb{E}_{p(f_i)}[f_if_i^\top]\end{aligned} \tag{3.44}$$

$$\int p(f_i|X_i,U)\log p(Y_i|f_i)df_i = -\frac{d}{2}\log(2\pi\beta^{-1}) - \frac{\beta}{2}Y_iY_i^\top + \beta K_{im}K_{mm}^{-1}UY_i^\top - \frac{\beta}{2}d\left(k_{i,i} - K_{im}K_{mm}^{-1}K_{mi}\right) - \frac{\beta}{2}K_{im}K_{mm}^{-1}UU^\top K_{mm}^{-1}K_{mi} \quad (3.45)$$

K_{mm}, K_{im} and $k_{i,i}$ are respectively the covariance matrix between the M inducing points, the covariance matrix between i_{th} input and M inducing points, and the covariance between input X_i and itself.

$$\log p(U|Z) = \log(2\pi)^{-nd/2} - \frac{d}{2}\log|K_{mm}| - \frac{1}{2}Tr(U^\top K_{mm}^{-1}U) \quad (3.46)$$

The variational distribution over latent variables is defined as

$$\log q(U) = \log\mathcal{N}(U|\mu_U,\Sigma_U) \quad (3.47)$$

In traditional variational sparse GP, the optimal form of mean and covariance matrix can be solved analytically by using calculus of variations and Lagrange multipliers[9]. In this way, it is assumed that all training data is available when updating parameters. This means the update can only happen after all training data are evaluated, namely, each iteration has to go through all training data points. Learning parameters, obviously, becomes very slow unless training dataset is very small. Stochastic variational inference is then proposed to deal with the scalability of variational inference.

Stochastic Variational Inference

As datasets become increasingly large, the scalability of variational inference is a key factor to its value in practice. For previous example, if both μ_U and Σ_U are treated as variational parameters, they can be learned during optimisation. There is no need to derive their expression analytically. As Σ_U needs to be positive definite, it can be written in the following form

$$\Sigma_U = WW^\top + diag(c)$$

introducing two training parameters. Therefore, μ_U, together with W and c are learned and updated as the model trains through iterations. In other words, they can be updated with estimates given subsets of the training data. Therefore, in each iteration the training only

evaluate a subset of complete training data to give estimate of the parameters. This process repeats until all training data are evaluates.

This method is called stochastic variational inference[11, 12] and this is the method adopted in the implementation of RGP in this thesis. Compared to traditional variational inference, a large part of derivation is replace by direct model training. It also has the ability to deal with large dataset unlike traditional variational inference. In traditional variational inference parameter updates can only happen after evaluating the complete training data in each iteration, which proves to be extremely slow as datasets become larger. The advantage of statistical variational inference is that the parameters can be updated after seeing some subset of data and this makes training on big dataset much more efficient.

Chapter 4

Experiments and Discussion

In this chapter, different experiments are conducted to explore the representation power of RGPs on human motion modelling with several variations in architecture. There are two main variations investigated. The first one is a fully observed model, which is a RGP with no hidden layer. This structure is also known as GP-NARX. The other is a RGP with one latent layer, called latent model. Unlike GP-SSMs, this architecture has recurrent structure in both hidden and observation layers. Both architectures are experimented on toy data and Mocap dataset. Walking and running are the two main motions the modelling is focused on. Golf and jumping motions are also tested on the same model after the successful modelling of walking and running sequences.

4.1 Experiment Set-up

The experiments are run on GPU with Azure data science virtual machines. The library used for building the model is MxNet and GPy. MxNet can perform automatic differentiation useful for deriving the objective function. GPy is used to process the data. The data used are downloaded from CMU motion capture data website [1]. Jupyter notebooks containing all experiments are available at https://github.com/MaggieWYZW/humanmotionRGP.git.

4.1.1 Fully-observed Model

A fully observed one layer RGP is shown in Fig 4.1, where x is the control signal and y is the observation sequence. In a model without control signal, current observation only depends on the previous observations. Fig 4.1 shows the case where the observation is dependent on

one previous time step. However, the model can expand the dependency window making current observation dependent on more than one observation from previous time steps. The same principle goes to control signal, where current observation can be controlled by current control signal as well as multiple previous control signals. This arrangement gives the model the ability to consider longer term dependencies which could be potentially beneficial to long-term predictions.

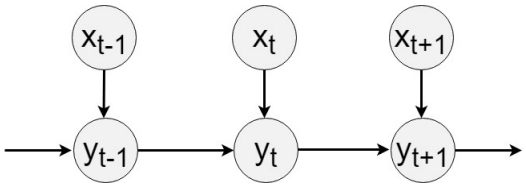


Fig. 4.1 Fully-observed Model with Control signal

The window size of control signal and history can vary. In Fig 4.1, the window size for both signal are 0, meaning only current control signal and previous one step are considered for predicting current observation. However, the window size in experiments on Mocap dataset is chosen to be 20 to allow longer history information to contribute to current prediction.

Human skeletons in Mocap data are structured in 29 joints. As the fully observed model uses the motion sequence directly, it can be used to model each joint independently. On the other hand, the model can also be varied to take into account the dependencies between joints. Considering skeleton natural structure and possible variations of the RGP fully-observed model, four different experiments are conducted making small variations on the baseline model described above. The details of the models and results are listed in later sections within this chapter.

4.1.2 Latent Model

Unlike fully observed models, where each joint can be modelled independently, the latent RGP requires a collective model for all body parts. Because the latent layer is used to summarise high dimensional human motion sequences to a low dimension. In other words, the 62 dimensional skeleton representation is used to train one model where all the dimensions in the observation sequence are generated from a lower dimensional latent space. For the observation layer, there are two configurations shown in Fig 4.2 where the observation is either solely dependent on latent representation or dependent on both latent layer and previous observation sequence.

The experiments are conducted with the architecture in Fig 4.2(a), where the history of both hidden layer and observation are used in predicting motion at current time step. Similar to previous fully-observed model, the latent model also has the flexibility to include or exclude control signal.

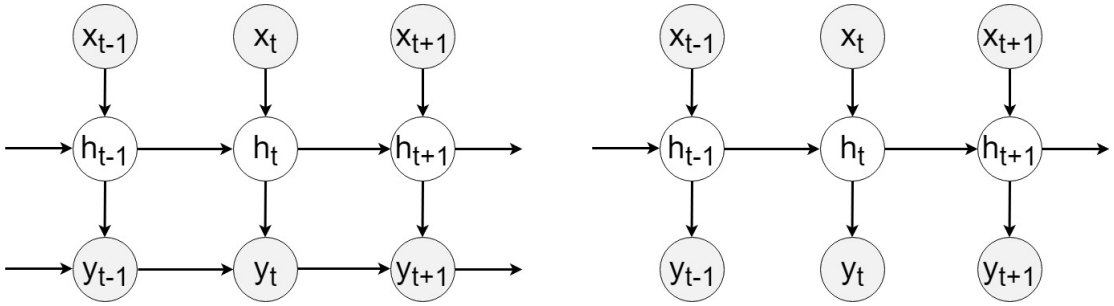


Fig. 4.2 Model with one hidden layer, where shaded nodes are observed and non-shaded is latent. (a) has recurrent relations in the observation layer and latent layer, meaning the observation sequence depends both on the latent representation and previous motion. In (b) the observation sequence only depends on the latent representation.

4.2 Experiments with Toy Examples

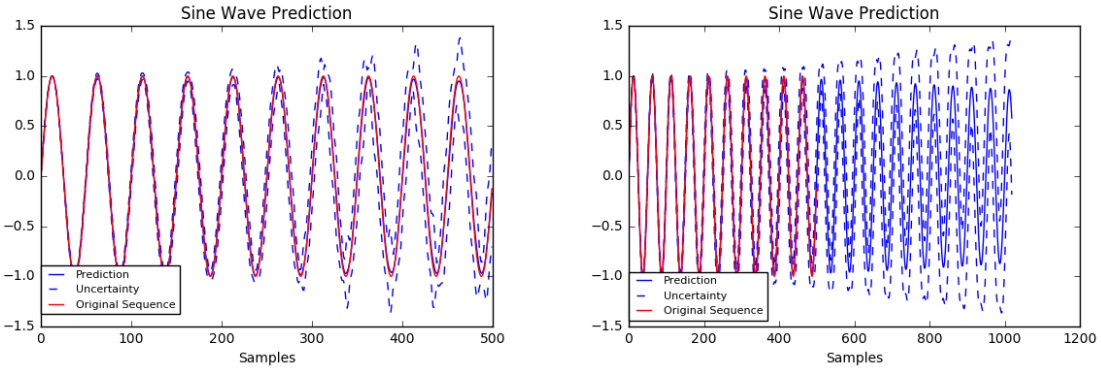
A very simple sine wave is used as toy data to test the performance of the RGPs. The sine wave contains 200 samples with frequency of 5Hz and sampling rate of 20. The test sine wave has the same frequency as training sequence and only an initial window of 20 samples are needed to predict the waveform.

4.2.1 Fully-observed Model

The fully observed model without control signal is tested on a sinusoidal waveform. In this toy example, the sine wave is the observation sequence and the model is trained on the observation sequence only. The history window size is 20 samples.

As the model output is probabilistic, the predictions are generated by taking the average of 100 samples from the output. Fig 4.3a shows that the model generates sine wave that matches the testing waveform well, with red line being the ground truth and blue being the prediction. The dashed lines are calculated by computing the variance of 100 samples. Naturally, as prediction goes further into the future more uncertainty is displayed at the

output. This phenomenon becomes more obvious as the model continues to predict longer sequences, as shown in Fig 4.3b. Nonetheless, the average perfectly predicts the frequency of the waveform, with the amplitude of the waveform decreasing slightly.

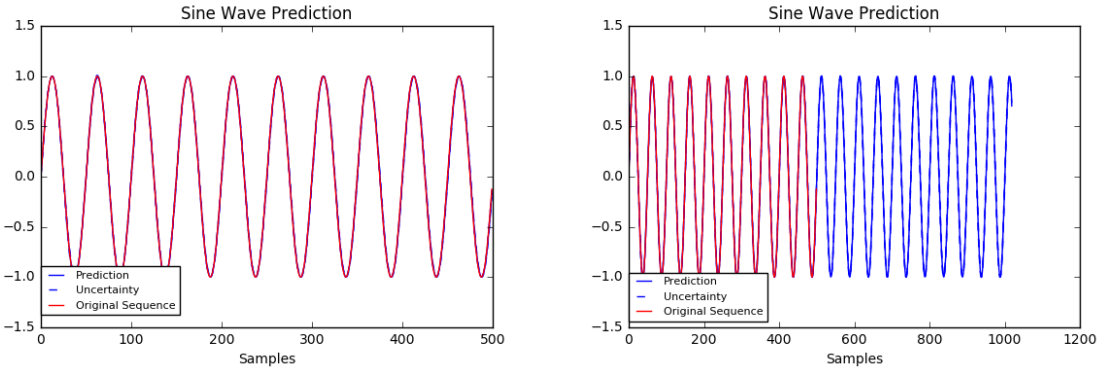


(a) (b)

Fig. 4.3 (a) Prediction of Sinusoidal waveforms using fully-observed RGP model. (b) is prediction of a long sequence.

4.2.2 Latent Model

The latent model without control signal is used to test on the toy data, where one hidden layer is added to the original fully-observed RGP. The window length is still 20 samples.



(a) (b)

Fig. 4.4 (a) Sine wave prediction of original sequence length. (b) is prediction of a long sequence.

The experiment results are shown in Fig 4.4. Fig 4.4a shows little uncertainty in the prediction compared to fully-observed model, so does Fig 4.4b when prediction horizon is

doubled. The reason can be that the expressive power of a latent model is more than sufficient for modelling a simple sine wave. The model fully captured and learnt the features in the data during training. From the initial results of the toy data, it can be concluded that latent model has more representation power over simple fully-observed model.

4.3 Experiments with Mocap Data

4.3.1 Fully-observed Models

It is natural that the joints of a person have structured correlations. In Mocap dataset, the joints of a skeleton follow certain hierarchy as shown in Fig 4.5. Each joint is represented with a local degree of freedom based on its parent joint. For example, *'thorax'* is dependent on *'upperback'*. The only special joint is *'root'* having a global coordinate and angle representation. Given this structure, several variations of the fully-observed model is explored to find a more suitable model for producing natural human motions.

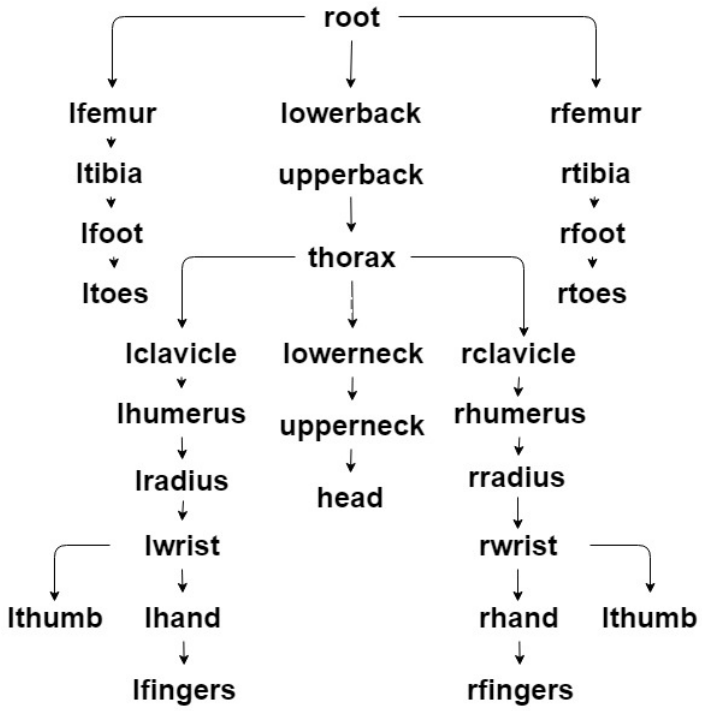
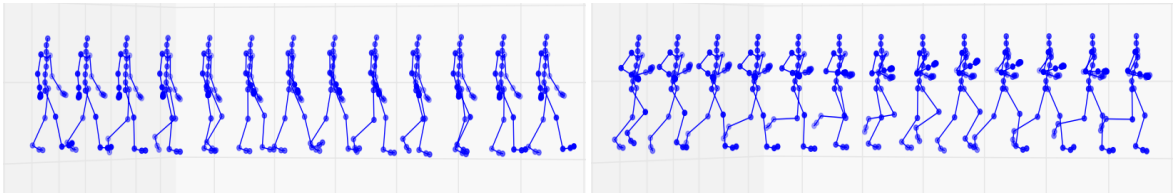


Fig. 4.5 Skeleton Hierarchical Structure

The motions to be modelled are walking and running sequences. The original test motion sequences are shown in Fig 4.6 as ground truth, where Fig 4.6a shows the walking motion

and Fig 4.6b shows the running motion. All prediction results are rendered into videos available at https://drive.google.com/open?id=1J38DbMuSAb_UEHUEcl9ea3GIo1qKYqF3.



(a) (b)

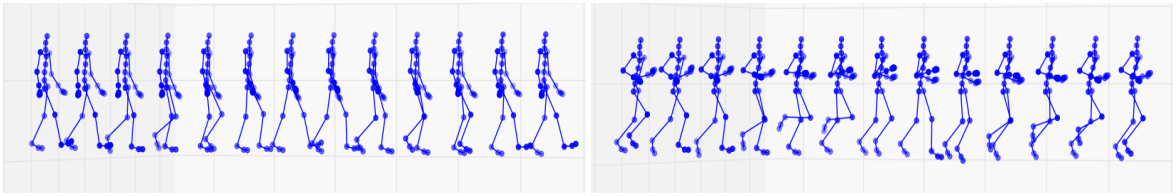
Fig. 4.6 (a) is the original test walking sequence. (b) is the original test running sequence.

Variation One: Fully-observed Independent Models without Control Signals

A fully-observed independent model is one that doesn't consider the dependencies between joints. Namely, each joint forms its own RGP independent from each other. The joint probability of the motion can therefore be factorised into the multiplication of each joint probability shown in Equation 4.1. As each joint has much fewer dimension compared to the whole skeleton, it is considered easier to model independently. For example, joint *'root'* $p(y^1)$ has its own covariance matrix and so does every other joint.

$$p(\mathbf{y}) = p(y^1)p(y^2)...p(y^{29}) \tag{4.1}$$

For prediction, the initial 20 frames of the test sequence are fed into the trained model for motion generation. The prediction sequence length is the length of the testing sequence.

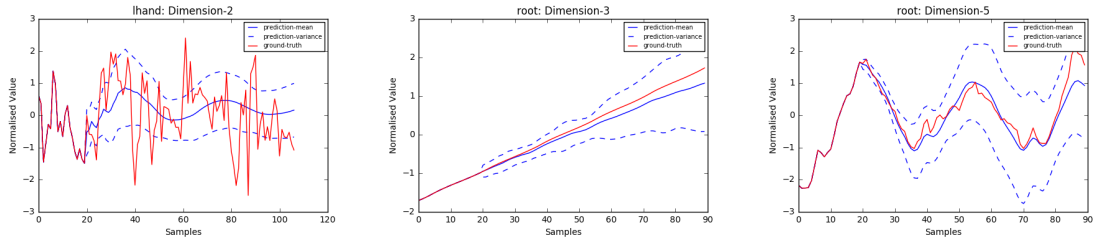


(a) (b)

Fig. 4.7 Synthesised human motions. (a) is generated walking sequence, and (b) is the generated running sequence.

From the results shown in Fig 4.7, the model can capture the basic movements of the walking and running motion. The most obvious differences between these two motions are the arm swing and feet position. Walking has a small range of arm swing and both feet always stay on the ground, whereas running has a large range of arm swing and one of the

feet is in the air during every leap forward. The synthesised walking motion is recognisable although not perfect, but the synthesised running motion has uneven step length on each foot step resulting in limping motion.



(a) (b) (c)

Fig. 4.8 Prediction for Single Dimensions of walking motion with fully-observed independent model without control signal. (a) is one of the dimensions from hand joint, which displays very high frequency content with less periodic feature. (b) is the third dimension of the *'root'* joint which is linear throughout time. (c) is the 5th dimension of the *'root'*, which has the most common periodicity among all joint dimensions.

There are three typical types of waveforms among all joints as shown in Fig 4.8. Fig 4.8c shows the most common type of waveform, a somewhat periodic waveform that is expected from walking cycles. Fig 4.8b shows a linear line indicating the body position moving forward. Fig 4.8a and Fig 4.8c have similar overall periodicity, but the waveform in Fig 4.8a has high frequency content. Evidently, the model fails to capture the high frequency details in some dimensions like feet and hands despite the global periodicity. This explains the results of the overall synthesised motion with missing or incorrect details.

Variation Two: Fully-observed Correlated Model with Control Signal

The fully-observed correlated model refers to one that considers the dependencies of joints in a skeleton, where each joint is dependent on its parent. In implementation of such a model, each joint model has its parent joint as control signal input. There is only one overall control signal, which is the delta of *'root'* joint that can be computed from

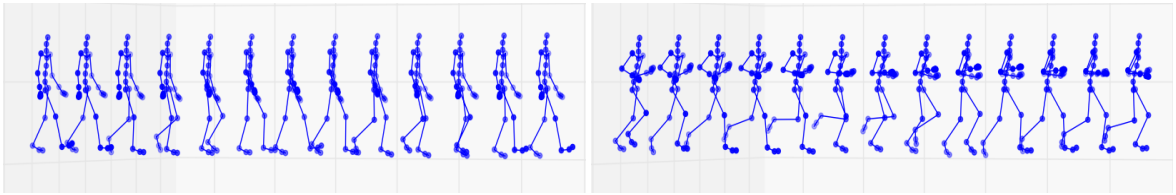
$$\mathbf{u} = \Delta = [0, y_2 - y_1, ..., y_t - y_{t-1}, ...y_N - y_{N-1}] \tag{4.2}$$

The delta computed is a 6-dimensional vector sequence. The overall model of the whole skeleton can be written as

$$p(\mathbf{y}) = p(y^1|\Delta^1)p(y^2|y^1)...p(y^{29}|y^{28}) \tag{4.3}$$

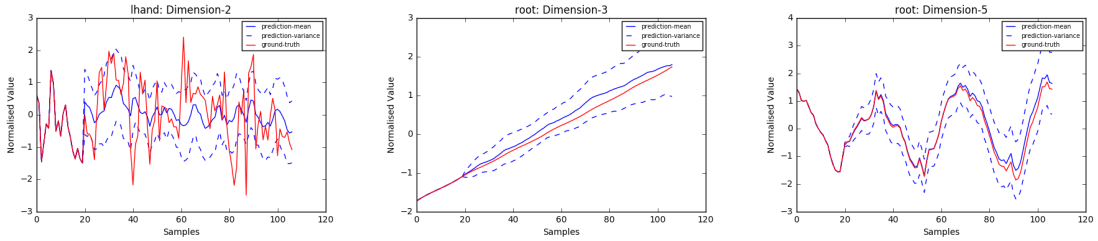
In this case, each joint is conditionally independent from each other given its parent joint. The window size for both control signal and observation sequence is 20 frames.

$$\bar{y}_i = [y_i, ...y_{i-19}]^\top, \quad \bar{u}_i = [u_i, ...u_{i-19}]^\top \tag{4.4}$$



(a) (b)

Fig. 4.9 Fully-observed model with correlated joint and control signal. (a) walking, (b) running.



(a) (b) (c)

Fig. 4.10 Prediction for Single Dimensions of walking motion with fully-observed correlated model with delta control signal. (a) is one of the dimensions from *'hand'* joint, which displays very high frequency content with less periodic feature. (b) is the third dimension of the *'root'* joint which is linear throughout time. (c) is the 5th dimension of the root, which has the most common periodicity among all joint dimensions.

On the single dimension level, improvements can be seen on three different types of waveforms in Fig 4.10. Fig 4.10a shows that the *'hand'* joint is modelled with more high frequency details. Both Fig 4.10b and Fig 4.10c shows better prediction compared to ground truth and lower uncertainty.

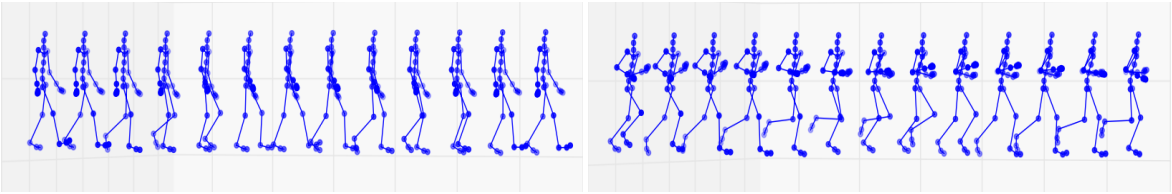
At the complete motion level, similar results are observed for generated walking sequences despite some foot slip and glitch on body position. Unfortunately, this model did not improve the running motion generation as expected. It resolved the problem in **Variation One**, that two sides of the body have uneven motions, but new problem arose. As prediction goes further into the future, the step size and leg swing become so small that it fails to

represent a running step. This may be due to the conditional independences of each joint. The joint further away from *'root'* are less aware of the global body position. This inspired the next variation where the delta control signal for *'root'* is applied to all joints.

Variation Three: Fully-observed Correlated Model with Full Control Signal

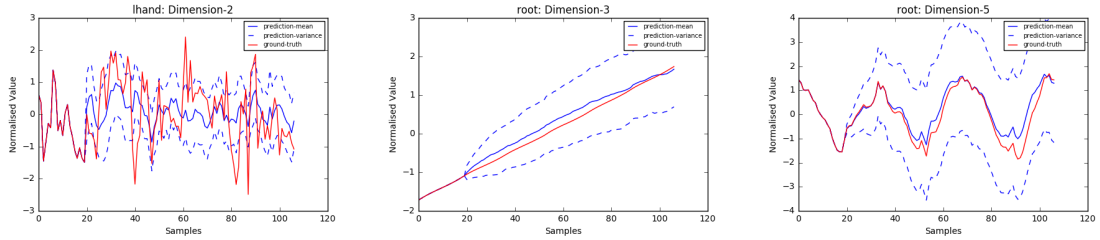
The fully-observed correlated model with full control signal considers the dependencies between joints, the same method presented in **Variation Two**. The full control signal means each joint model not only depends on its parent, but also has control signal as input which is the global delta of the *'root'* joint.

$$p(\mathbf{y}) = p(y^1|\Delta^1)p(y^2|y^1,\Delta^1)...p(y^{29}|y^{28},\Delta^1) \tag{4.5}$$



(a) (b)

Fig. 4.11 Fully-observed model with correlated joints and control signal on all joints. (a) generated walking sequence, (b) generated running sequence.



(a) (b) (c)

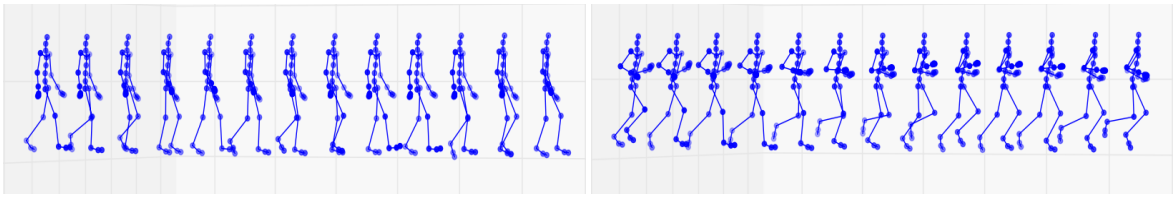
Fig. 4.12 Prediction for Single Dimensions of walking motion with fully-observed correlated model with control signal on all joints. (a) is one of the dimensions from *'hand'* joint, which displays very high frequency content with less periodic feature. (b) is the third dimension of the *'root'* joint which is linear throughout time. (c) is the 5th dimension of the *'root'*, which has the most common periodicity among all joint dimensions.

This model variation shows similar synthesis results for both walking and running motions. Firstly, the single dimension prediction results in Fig 4.12 show similar prediction mean

values but higher uncertainty. This may be due to additional control signal for each joint model. Provided the motion is generated from the mean values of the samples, the predicted motions do not vary much. However, the larger uncertainty in this model means that it can incorporate more variations from the original training motion, which is potentially beneficial for tracking tasks.

Variation Four: Mixed Model

If one model is able to capture and generate multiple motions, it will save a lot of training time. In order to test the representation power of the RGP model on multiple motions, both walking and running sequences are trained on one model. The architecture of the model used here is the same as **Variation Three**. After the model is trained, it is used to predict walking and running sequences independently.



(a) (b)

Fig. 4.13 Fully-observed model with correlated joints and control signal on all joints trained on both walking and running sequences. (a) generated walking sequence, (b) generated running sequence.

Qualitatively, few differences in modelling results are shown across different variations. To compare the results quantitatively, mean squared error(MSE) of testing sequences are computed, shown in Table 4.1. Compared to benchmark model, **Variation One**, all other variations have some improvements based on MSE values of each test sequence. The synthesised motions show obvious features of walking and running. However, the problem of foot slip is amplified in the mixed modelling where the skeleton, at times, looks like it is sliding on ice. This can hardly be observed from Fig 4.13 but very obvious in the video rendered from motion files. Reasons for the performance may be that in some joints the difference between walking and running motion is not very significant, therefore, the model fails to learn two different dynamics. Looking at the local values at a global perspective, they can both influence the movement of the skeleton making the whole movement coherent. Considering the highly structured nature of human motions, a hidden layer may be able

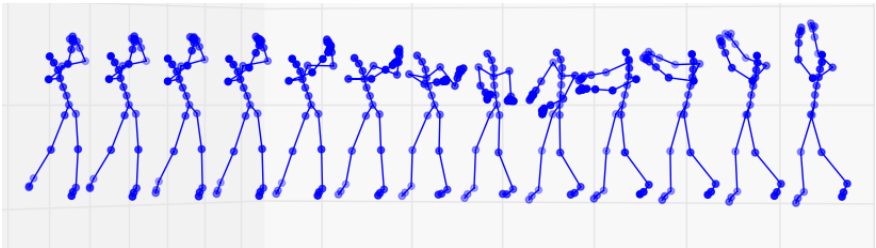
to capture high level structures in the latent space. The latent model and experiments are presented in section 4.3.2.

Model & Variations		Walk				Run			
		seq1	seq2	seq3	seq4	seq1	seq2	seq3	seq4
Fully-observed Model	Variation One	0.458	0.432	0.405	0.443	0.413	0.492	0.416	0.424
	Variation Two	0.366	0.274	0.280	0.357	0.439	0.481	0.353	0.389
	Variation Three	0.408	0.269	0.298	0.357	0.466	0.550	0.406	0.431
	Variation Four	0.387	0.307	0.266	0.343	0.380	0.347	0.367	0.370
Latent Model	Variation One	0.337	0.245	0.308	0.311	0.347	0.346	0.303	0.335
	Variation Two	0.365	0.260	0.315	0.341	0.357	0.342	0.318	0.347
	Variation Three	0.339	0.260	0.272	0.355	0.330	0.305	0.296	0.343

Table 4.1 MSE of prediction sequences

Golf

(a)



(b)

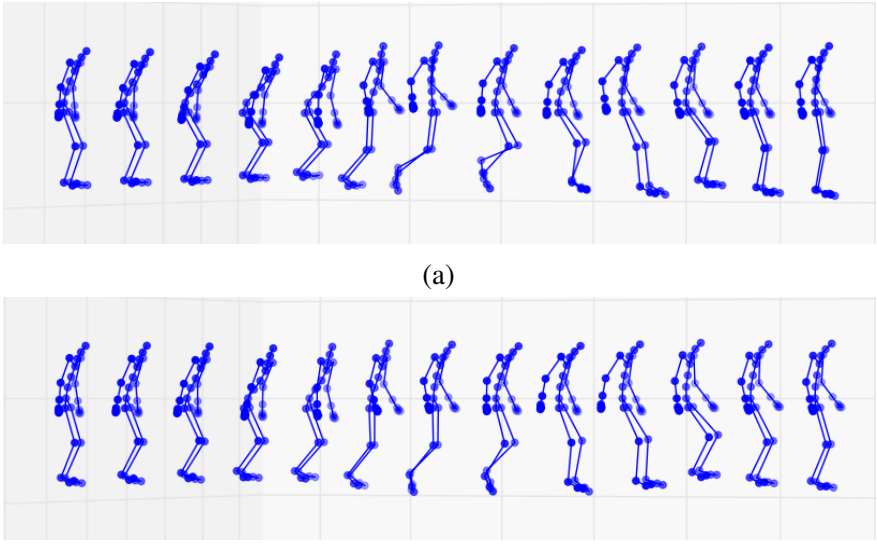
Fig. 4.14 Fully-observed model with delta control on all joints for golf motion. (a) is the ground truth of the golf motion, (b) is generated golf sequence.

Variation Four of fully-observed model is used to model golf swing motion. Golf is very different from walking and running where the movement range is similar in four limbs. It has small range of movements in the lower body but large swing movements in the arms. Nevertheless, the model successfully modelled golf swing motion, despite of some undesired

body rotation, showing that it is able handle large range of movements in human motion modelling. The synthesised swing movement is shown in Fig 4.14b, and the original golf motion sequence is in Fig 4.14a.

Jumping

Another motion, jumping, is also trained and tested using the same model as golf motion modelling. The generated skeleton is able to jump forward with compression at the start, pause in the air and landing motion. The three stages can be seen from Fig 4.15b. However, some details in the body and arm joints are missing from the original jumping motion in Fig 4.15a.



(a)

(b)

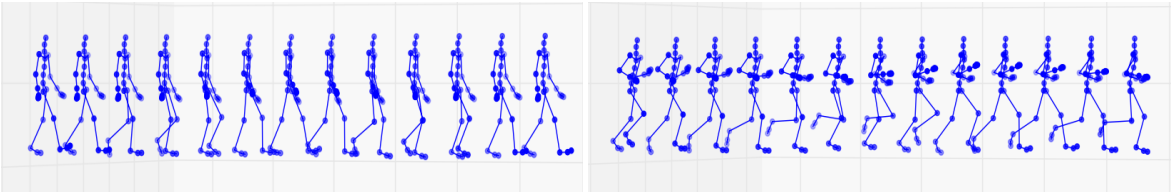
Fig. 4.15 Fully-observed model with delta control on all joints for jumping motion. (a) is the ground truth of the jump motion, (b) is generated jumping sequence.

4.3.2 Latent Models

The latent models used in the experiments have one hidden layer between input signal and observation sequence i.e. motion sequence. The structure implemented is shown in Fig 4.2a where current motion is dependent on latent history and observation history. Latent models treat a human skeleton as a whole and train one model that is able to generate values for all joints.

Variation One: One Hidden Layer without Control Signal

The first variation of the latent model does not have control signal as input, thus only having two layers in the graphical representation. This variation test the representation power of latent model without external influence. In this case, the latent nodes only depends on information passed from its own history.



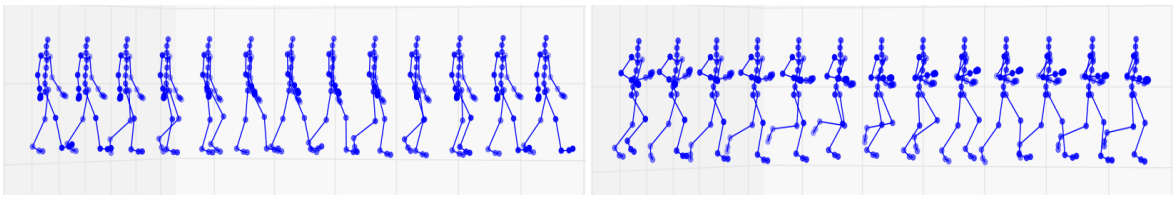
(a) (b)

Fig. 4.16 Latent model without control signal (a) generated walking motion, (b) generated running motion.

Quantitatively and qualitatively, the synthesised motions by latent model behave more naturally than fully-observed models. One of the biggest improvement is the motion decay throughout time, where the step size and movements remain in the same range even as the prediction goes further into the future. Foot slip, one of the most significant problem in previous fully-observed models, also improved in the latent model. It is observed that the body position and body parts movements are more in sync and the foot slip between steps is greatly reduced.

Variation Two: One Hidden Layer with Control Signal

In addition to **Variation One**, this model has a control signal as input to the latent layer. The control signal used in this experiment is the same as in the fully-observed model, see Equation 4.2. The delta control value is computed from the 6-dimensional *'root'* global coordinates and angles.



(a) (b)

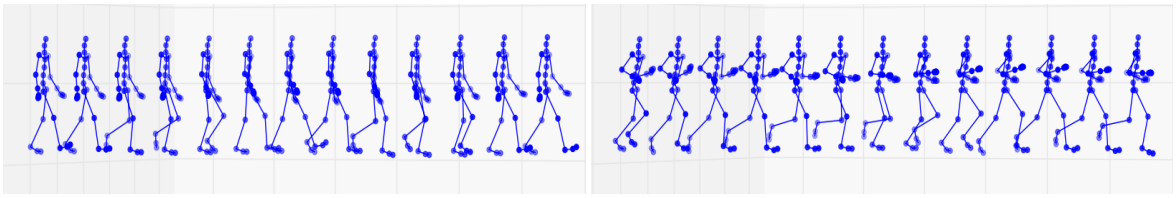
Fig. 4.17 Latent model with control signal (a) generated walking motion, (b) generated running motion.

The generated motions are shown in Fig 4.18. The synthesised motion sequences perform similar to those without control signals, showing that introducing control signal in this scenario does not improve the performance.

There are a few explanation for this result. On the one hand, the control signal is input to the hidden layer rather than directly inputted to the observation layer like in the fully-observed model. This may cause the input to have less influence on the observation sequence. On the other hand, the control signal has 6 dimensions and observation layer has 62 dimensions. When the 1-dimensional hidden layer is trying to capture the features from both the 62-dimensional space and 6-dimensional space, the higher dimension is more dominant leading to the input signal having little influence on the output.

Variation Three: One Hidden Layer Mixed Model

The third variation of latent model uses the same model as in **Variation Two**, but trained on both walking and running sequences. In order for the model to represent the differences between different models, the global mean and variance across all sequences are used to normalise both walking and running data.

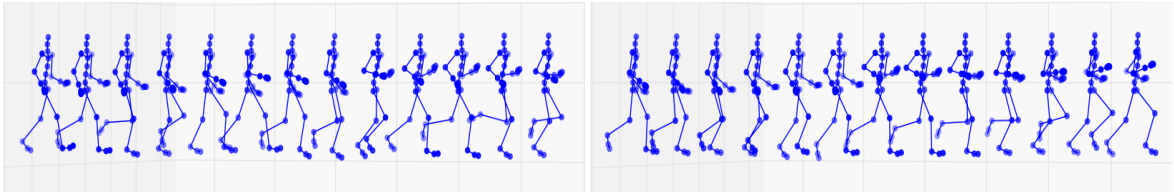


(a) (b)

Fig. 4.18 Latent model with control signal trained on both walking and running sequences. (a) generated walking sequence, (b) generated running sequence.

The prediction from the trained model gives satisfying results on both walking and running motions. The skeleton shows the most natural motions among all other models with accurate foot placement and body position. The motion no longer looks like it is sliding on ice. For running motion, the skeleton can leap forward evenly on both sides and have a reasonable step size as it leaps forward.

Fig 4.19 shows the prediction from the model when a skeleton is in transition from a walking to running motion. To control the motion, the delta of the walking and running sequence are concatenated together for controlling the transition. To recover the motion to the correct scale, different means and variances are extracted from walking and running sequences. The motion can be separated into three sections: walking, transition and running. Walking and running sections are recovered using their corresponding mean and variances, while transition period is rescaled using the average of the mean and variance from the walking and running sequences.



(a) (b)

Fig. 4.19 Latent model with control signal trained on both walking and running sequences. (a) transition from walking to running with initial window as walking, (b) transition from walking to running with initial window as running.

Depending on the initial window, the two sequences have different behaviours. When the initial window is in walking motion, the overall body position is based on walking. This leads to short step distance when the skeleton starts to run. The opposite is observed in Fig 4.19b when the initial window is in running motion. At the start of the motion sequence the skeleton has walking motions but it is also sliding 'on the ground', as the step size for running is larger.

The number of dimensions for the hidden layer is also tested and the MSEs for each dimension are shown in Table 4.2. From the results, increasing the dimensionality of the latent layer does not guarantee an improvement in performance. Therefore, a simpler model is preferred for the same performance.

Models	HD	Walk				Run			
Variation One:	1	0.337	0.245	0.308	0.311	0.336	0.36	0.297	0.339
Latent Model	2	0.367	0.235	0.305	0.315	0.347	0.346	0.303	0.335
without Control	3	0.361	0.259	0.306	0.333	0.353	0.335	0.311	0.342
Variation Three:	1	0.351	0.256	0.266	0.334	0.338	0.318	0.306	0.340
Latent Mixed	2	0.339	0.260	0.272	0.355	0.327	0.311	0.296	0.343
Model	3	0.352	0.260	0.277	0.328	0.326	0.313	0.294	0.322

Table 4.2 MSE of prediction sequences with varying hidden layer dimensions (HD: hidden layer)

Golf

A latent model without control signal is used to generate golf motion. The motion sequence shown in Fig 4.20 is smoother than produced from fully-observed model. The skeleton seems to be able to present more structures behind the movements. For example, at the end of the swing the skeleton has one foot stay flat on the ground and the other one tip-toe as both hands swing above the head. The fully-observed model fails to capture details like this. However, it is the details that make the motions more natural and realistic.

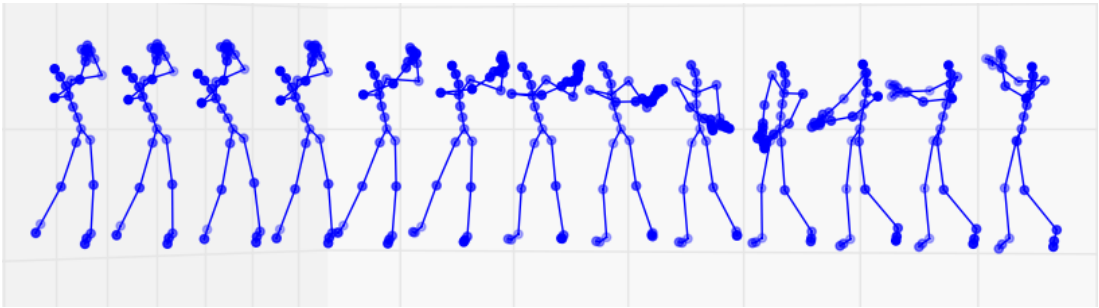


Fig. 4.20 Latent model without control: Golf

Jumping

A latent model with control signal is used to generate jumping motions. The control signal is used to control the height of the jump. As the upright direction has the most drastic value fluctuation among all joint dimensions, a control signal is used to amplify the influence of that particular dimension. The resulting motions are shown in Fig 4.21. There is also reduced sliding before jumping and after landing.

Fig. 4.21 Latent model with control: Jumping

4.4 Summary

Overall, the latent model is able to generate more realistic motions. The hidden layer is able to capture some higher level structure of the skeleton and movements, therefore, providing more precise movement predictions. One of the most important improvement is the foot slip across all motions. After adding the hidden layer, the sliding is greatly reduced resulting in more natural motions. The latent model is also more robust as shown in latent mixed model experiment. The input signal has little, nearly none, impact on the generated signal given the initial window in prediction. This is a challenge to be further addressed as a model that has the flexibility to be controlled by an input signal is desirable. Therefore, there needs to be a balance between robustness and sensitiveness toward inputs signals.

Chapter 5

Summary and Conclusion

5.1 Future Work

Human motion modelling is a challenging task as it is hard to capture the complex skeleton structure, and have total flexibility to control the motions at the same time. This dissertation successfully presented models based on RGPs that are able to model human motions such as walking, running, golf and jumping. The results provide evidence for the high representation power of the model architecture of RGP, especially the latent model.

However, some issues still need to be addressed. First of all, the unnatural sliding of the skeleton during movements still remains a problem despite the good performance from latent model. The sliding is most obvious when transitioning from initial window to the first few frames of prediction, where the transition is less smooth. One possible solution to this problem is to consider the global position of the feet. If the skeleton can learn to fix on these positions, it is more likely to avoid sliding during movements.

Secondly, the control signal has less control over the movements than expected, especially in the latent model. Even though one model can generate more than one motion, the control signal cannot effectively tell the model to switch between different motions. This is limiting the flexibility and scalability of the model. One potential way to improve this could be to consider variations on the model architecture, for example feeding the control signal as an additional input to the observation layer, which provides a more direct way of influencing the motion itself. Another solution could be using mixtures of GPs[25]. The idea is that different GPs are used at some local regions for a mixture of different GP models. In other words, as the joints of a skeleton can be grouped based on its movement features, these groups

of dimensions can be modelled with different GPs jointly forming a mixture of Gaussian process model.

Finally, we can look at human motion modelling problem at a higher level, considering the human skeleton structures and inherent common features between motions. This leads to a better lower-dimensional representation extracted from the movements. Additionally, the lower-dimensional data can be used to train similar autoregressive models. As a result, it requires the predictions to be remapped to the original higher dimension space. The control signal can be applied directly to the lower dimension space, which will determine the motions in the higher dimension.

5.2 Conclusions

In this dissertation, the problem of 3D human motion modelling is discussed. The focus is on skeleton structured 3D motion representation for motion synthesis. The different types of models in the literature for this modelling task are also reviewed with particular focus on model variations on Gaussian processes. The dissertation also dives into details of Gaussian processes and approximation methods such as variational inference and sparse Gaussian processes. Especially, stochastic variational inference is introduced and implemented in the modelling. Moreover, sparse GP is also non-trivial when it comes to elegant representation and efficient computation. Furthermore, several experiments are presented and the results are discussed on different model variations based on RGP.

The final synthesised motions from the latent model show the great representation power of the RGP model. The comparison is made in both quantitative and qualitative manner, where the latent models outperform the rest. The positive results open doors for future improvements on the human motion modelling and possible future applications.

References

[1] Carnegie mellon university - cmu graphics lab - motion capture library. URL http://mocap.cs.cmu.edu/info.php.

[2] A. Bissacco, A. Chiuso, Yi Ma, and S. Soatto. Recognition of human gaits. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–52–II–57 vol.2, 2001. doi: 10.1109/CVPR.2001.990924.

[3] Judith Bütepage, Michael Black, Danica Kragic, and Hedvig Kjellström. Deep representation learning for human motion prediction and classification. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, Piscataway, NJ, USA, July 2017. IEEE.

[4] Andreas Damianou. Deep gaussian processes and variational propagation of uncertainty. *PhD Thesis, University of Sheffield*, 2015.

[5] Andreas C. Damianou and Neil D. Lawrence. Deep gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, pages 207–215, 2013. URL http://jmlr.org/proceedings/papers/v31/damianou13a.html.

[6] Andreas Doerr, Christian Daniel, Martin Schiegg, Duy Nguyen-Tuong, Stefan Schaal, Marc Toussaint, and Sebastian Trimpe. Probabilistic recurrent state-space models. In *Proceedings of the International Conference on Machine Learning*, July 2018.

[7] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 4346–4354, Washington, DC, USA, 2015. IEEE Computer Society. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.494. URL http://dx.doi.org/10.1109/ICCV.2015.494.

[8] Roger Frigola, Yutian Chen, and Carl E. Rasmussen. Variational gaussian process state-space models. *CoRR*, abs/1406.4905, 2014. URL http://arxiv.org/abs/1406.4905.

[9] Yarin Gal and Mark van der Wilk. Variational inference in the Gaussian process latent variable model and sparse GP regression – a gentle tutorial. *arXiv:1402.1412*, 2014.

[10] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. *CoRR*, abs/1704.02827, 2017. URL http://arxiv.org/abs/1704.02827.

[11] James Hensman, Nicoló Fusi, and Neil D. Lawrence. Gaussian processes for big data. *CoRR*, abs/1309.6835, 2013. URL http://arxiv.org/abs/1309.6835.

[12] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, May 2013. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2502581.2502622.

[13] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35:138:1–138:11, 2016.

[14] James N. Ingram, Konrad P. Körding, Ian S. Howard, and Daniel M. Wolpert. The statistics of natural hand movements. *Experimental Brain Research*, 188(2):223–236, Jun 2008.

[15] Ashesh Jain, Amir Roshan Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. *CoRR*, abs/1511.05298, 2015. URL http://arxiv.org/abs/1511.05298.

[16] Wessel Bruinsma Richard E. Turner James Requeima, Will Tebbutt. The gaussian process autoregressive regression model. In *ARXIV*, 2018.

[17] Neil Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *In NIPS*, page 2004, 2003.

[18] Tsungnan Lin, B. G. Horne, P. Tino, and C. L. Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6): 1329–1338, Nov 1996. ISSN 1045-9227. doi: 10.1109/72.548162.

[19] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, Piscataway, NJ, USA, July 2017. IEEE.

[20] César Lincoln C. Mattos, Zhenwen Dai, Andreas Damianou, Jeremy Forth, Guilherme A. Barreto, and Neil D. Lawrence. Recurrent gaussian processes. In Hugo Larochelle, Brian Kingsbury, and Samy Bengio, editors, *Proceedings of the International Conference on Learning Representations*, volume 3, Caribe Hotel, San Juan, PR, 00 2016.

[21] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.

[22] Vladimir Pavlovic, James M. Rehg, and John Maccormick. Learning switching linear models of human motion. pages 981–987, 2000.

[23] Joaquin Quiñonero Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *J. Mach. Learn. Res.*, 6:1939–1959, December 2005. ISSN 1532-4435.

[24] Liva Ralaivola and Florence d'Alché Buc. Dynamical modeling with kernels for nonlinear time series prediction. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pages 129–136, Cambridge, MA, USA, 2003. MIT Press. URL http://dl.acm.org/citation.cfm?id=2981345.2981362.

[25] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[26] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. MIT Press, 2006. URL http://papers.nips.cc/paper/2857-sparse-gaussian-processes-using-pseudo-inputs.pdf.

[27] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. MIT Press, 2006.

[28] Yongyi Tang, Lin Ma, Wei Liu, and Wei-Shi Zheng. Long-term human motion prediction by modeling motion context and enhancing motion dynamic. *CoRR*, abs/1805.02513, 2018. URL http://arxiv.org/abs/1805.02513.

[29] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In David van Dyk and Max Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 567–574, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR. URL http://proceedings.mlr.press/v5/titsias09a.html.

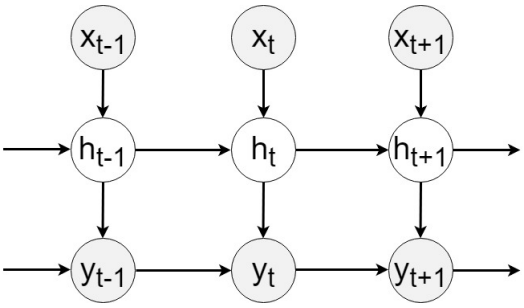
[30] Jack Wang, Aaron Hertzmann, and David J Fleet. Gaussian process dynamical models. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1441–1448. MIT Press, 2006.

[31] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):283–298, February 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1167.

[32] Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In *Advances in Neural Information Processing Systems 8*, pages 514–520. MIT press, 1996.

Appendix A

RGP Derivation



$$\begin{aligned} h_t &= f(\bar{x}_t, \bar{h}_{t-1}) + \varepsilon_f \\ g_t &= g(\bar{h}_t, \bar{y}_{t-1}) + \varepsilon_g \end{aligned} \tag{A.1}$$

where

$$\begin{aligned} \bar{x}_t &= [x_t, \ldots, x_{t-L_x+1}] \\ \bar{h}_t &= [h_t, \ldots, h_{t-L_h+1}] \\ \bar{y}_t &= [y_t, \ldots, h_{t-L_y+1}] \end{aligned} \tag{A.2}$$

For simplicity, assume $L_x = L_h = L_y = L$.

The log marginal likelihood can then be written as the following expression along with the definition for evidence lower bound.

$$\begin{aligned}
&\log p(Y|X) \\
&= \int p(Y|g,u_g,H,u_f,X)p(g|u_g,H)p(H|f,u_f,X)p(f|u_f,X)p(u_f)p(u_g)d(u_f,u_g,f,g,H) \\
&\geq \int Q \log \frac{p(Y|g,u_g,H,u_f,X)p(g|u_g,H)p(H|f,u_f,X)p(f|u_f,X)p(u_f)p(u_g)}{Q} d(u_f,u_g,f,g,H) \\
&:= \mathcal{L}
\end{aligned} \tag{A.3}$$

where the variational distribution Q is

$$Q = p(f|u_f,X)p(g|u_g,H)q(u_f)q(u_g)q(H) \tag{A.4}$$

Considering a mean-field approximation, each term in is given by the following:

$$\begin{aligned}
p(Y|g,u_g,H,u_f,X) &= \prod_{t=L+1}^{N} p(y_t|g_t,u_g,\bar{h}_t,\bar{y}_{t-1}) \\
&= \prod_{t=L+1}^{N} \mathcal{N}(y_t|g_t,\sigma_y^2 I)
\end{aligned} \tag{A.5}$$

$$\begin{aligned}
p(g|u_g,H) &= \prod_{L+1}^{N} p(g_t|u_g,\bar{h}_t) \\
&= \prod_{L+1}^{N} \mathcal{N}(g_t|K_{gu_g}K_{u_g}^{-1}u_g, K_g g - K_{gu_g}K_{u_g}^{-1}K_{gu_g}^{\top}
\end{aligned} \tag{A.6}$$

K_{gu_g} is the covariance matrix between input point of g and its inducing points, K_{gg} is the covariance matrix between the inputs of g and K_{u_g} is the covariance matrix between inducing points.

$$\begin{aligned}
p(H|f,u_f,x) &= \prod_{t=L+1}^{N} p(h_i|f_i,u_f,\bar{x}_t) \\
&= \prod_{t=L+1}^{N} \mathcal{N}(h_t|f_t,\sigma_g^2 I)
\end{aligned} \tag{A.7}$$

$$
\begin{aligned}
p(f|u_f, X) &= \prod_{i=L+1}^{N} p(f_t|u_f, \bar{x}_t) \\
&= \prod_{i=L+1}^{N} \mathcal{N}(f_t|K_{fu_f}K_{u_f}^{-1}u_f, K_{ff} - K_{fu_f}K_{u_f}^{-1}K_{fu_f}^{\top}
\end{aligned}
\tag{A.8}
$$

K_{fu_f} is the covariance matrix between input point of f and its inducing points, K_{ff} is the covariance matrix between the inputs of f and K_{u_f} is the covariance matrix between inducing points.

$$p(u_f) = \mathcal{N}(u_f|0, K_{u_f}) \tag{A.9}$$

$$p(u_g) = \mathcal{N}(u_g|0, K_{u_g}) \tag{A.10}$$

Here u_f is dependent on the input positions(Z_f) of the inducing points and u_g is dependent on the input positions(Z_g) of its inducing points.

The variational distributions are :

$$q(u_f) = \mathcal{N}(u_f|m_{u_f}, \Sigma_{u_f}) \tag{A.11}$$

$$q(u_g) = \mathcal{N}(u_g|m_{u_g}, \Sigma_{u_g}) \tag{A.12}$$

$$q(H) = \prod_{t=L+1}^{N} q(h_t|\hat{h}_{t-1}, \hat{x}_t) = \prod_{t=L+1}^{N} \mathcal{N}(h_t|\mu_{ht}, \Sigma_{ht}) \tag{A.13}$$

In the above, m_{u_f}, Σ_{u_f}, m_{u_g}, Σ_{u_g}, μ_{ht} and Σ_{ht} are all variational parameters.

The covariance matrix of $q(u_f)$ and $q(u_f)$ need to satisfy the criteria to be positive definite. Therefore, they can be written as $\Sigma_{u_f} = W_f W_f^{\top} + diag(c_f)$ and $\Sigma_{u_g} = W_g W_g^{\top} + diag(c_g)$. This form gives the parameters to train:

$$W_f, c_f, W_g, c_g, m_{u_f}, m_{u_g}$$

As for $q(H)$, its mean and covariance matrix are both constructed as fully connected neural networks.

The log term in the lower bound is

$$\begin{aligned}
&\log \frac{p(Y|g,u_g,H,u_f,X)p(g|u_g,H)p(H|f,u_f,X)p(f|u_f,X)p(u_f)p(u_g)}{Q} \\
&= \log \frac{p(Y|g,u_g,H,u_f,X)\cancel{p(g|u_g,H)}p(H|f,u_f,X)\cancel{p(f|u_f,X)}p(u_f)p(u_g)}{\cancel{p(f|u_f,X)}\cancel{p(g|u_g,H)}q(u_f)q(u_g)q(H)} \\
&= \log \frac{p(Y|g,u_g,H,u_f,X)p(H|f,u_f,X)p(u_f)p(u_g)}{q(u_f)q(u_g)q(H)}
\end{aligned} \tag{A.14}$$

The log expressions for each term are shown as follows:

$$\log p(u_f) = \log(2\pi)^{-nd_f/2} - \frac{d_f}{2}\log|K_{u_f}| - \frac{1}{2}tr\big(u_f^\top K_{u_f}^{-1}u_f\big)\big) \tag{A.15}$$

$$\log p(u_g) = \log(2\pi)^{-nd_g/2} - \frac{d_g}{2}\log|K_{u_g}| - \frac{1}{2}tr\big(u_g^\top K_{u_g}^{-1}u_g\big)\big) \tag{A.16}$$

$$\log q(u_f) = \log(2\pi)^{-nd_f/2} - \frac{d_f}{2}\log|\Sigma_{u_f}| - \frac{1}{2}tr\big((u_f-m_{u_f})^\top\Sigma_{u_f}^{-1}(u_f-m_{u_f})\big) \tag{A.17}$$

$$\log q(u_f) = \log(2\pi)^{-nd_g/2} - \frac{d_g}{2}\log|\Sigma_{u_g}| - \frac{1}{2}tr\big((u_g-m_{u_g})^\top\Sigma_{u_g}^{-1}(u_g-m_{u_g})\big) \tag{A.18}$$

$$\log q(H) = \sum_{t=L+1}^{N}\log(2\pi)^{-nd_f/2} - \frac{d_f}{2}\log|\Sigma_t| - \frac{1}{2}tr\big((h_t-\mu_t)^\top\Sigma_t^{-1}(h_t-\mu_t)\big) \tag{A.19}$$

$$\tag{A.20}$$

$$\log p(Y|g,u_g,H,u_f,X) = \sum_{t=L+1}^{N}\log(2\pi)^{-nd_g/2} - \frac{d_g}{2}\log\sigma_y^2 - \frac{1}{2}\sigma_y^2(y_t-g_t)^\top(y_t-g_t) \tag{A.21}$$

$$\log p(H|f,u_f,X) = \sum_{t=L+1}^{N}\log(2\pi)^{-nd_f/2} - \frac{d_f}{2}\log\sigma_h^2 - \frac{1}{2}\sigma_h^2(h_t-f_t)^\top(h_t-f_t) \tag{A.22}$$