Bachelor thesis

Artificial Intelligence

Radboud University 

Application of Gaussian Processes to robot state estimation

Author:
Esra Bakker
s4613376

First supervisor:
dr. M. Hinne
Artificial Intelligence,
Radboud University
Nijmegen
m.hinne@donders.ru.nl

Second supervisor:
dr. P.L. Lanillos Pradas
Artificial Intelligence,
Radboud University
Nijmegen
p.lanillos@donders.ru.nl

Abstract

Lanillos & Cheng [8] introduced "a computational perceptual model based on predictive processing that enables any multi sensory robot to learn, infer and update its body configuration when using arbitrary sensors with Gaussian additive noise". Specifically, their algorithm works by first using Gaussian Processes (GP) regression to learn the sensor generative model; the mapping between the joint angles of the robot and the sensor values, and then uses this learned generative model to generate a prediction about the state of the robot. A well know problem with GPs is that they don't scale well with a large number of data point or a high number of dimensions. This may make this proposed approach difficult to use for complex robotic systems. In order to asses whether this is a significant concern and whether it can be mitigated I examined how well a variety of their model (using an Extended Kalman Filter) performs on a variety of different robotic systems of increasing complexity. In addition I tested a sparse Gaussian Processes model to investigate its potentia as a method to mitigate the poor scaling of a traditional GP for complex robotic systems.

Contents

Chapter 1

Introduction

Predictive processing, or predictive coding, is an influential approach in computational neuroscience [5]. Within this approach, in order to infer the state of the world, predicted sensory input is compared to incoming sensory input [8], the prediction error is then used to update our estimate of the world state. This notion has been influential in neuroscience. For example, predictive coding has been able to explain neural activity in the visual cortex [11]. However, its application in the domain of robotics has received little attention. Lanillos & Cheng [8] introduced "a computational perceptual model based on predictive processing that enables any multi sensory robot to learn, infer and update its body configuration when using arbitrary sensors with Gaussian additive noise"[8]. Specifically, their algorithm works by first using Gaussian processes regression to learn the sensor generative model; the mapping between the joint angles of the robot and the sensor values, and then uses this learned generative model to generate a prediction about the state of the robot. This prediction is then compared with incoming sensory values to estimate the state.

Gaussian Processes are flexible models which can be used to learn functions [12]. They do this by modeling the mean and covariance function of the variables along different dimensions. While they are powerful they do not scale well computationally, specifically they scale $O(n^3)$ [7] where n is the size of the data set, and the complexity with respect to $O(n^3p^3)$ where p is the number of outputs or tasks with respect to dimensions [7]. Thus, they become infeasible as data set size increases.

To allow Gaussian processes scale to larger data sets and dimensions a number of approaches have been proposed. One of the major approaches is to replace the standard Gaussian processes regression with sparse Gaussian processes regression [7]. This allows us to reduce the data set to a subset of induced variables which reduces the complexity to $O(nm^2)$ where m is the number of inducing variables [7]. Additionally, augmented training methods have been developed, such as Stochastic Variational Inference which allows

us to further reduce the complexity to $O(m^3)$ [7]. These two techniques combined allowed Hensman, Fusi & Lawrence, to train a Gaussian process on a data set of 800,000 data points with 1000 induced variables [7]. This ability to improve Gaussian process scaling may be useful, considering the model proposed by Lanillos & Cheng [8] uses a standard Gaussian process regression which may not scale well to more complex robotic systems. For example Atlas [1], designed by Boston Dynamics, has 28 degrees of freedom, something which traditional Gaussian process regression may not be feasible for.

The primary goal of this project is to asses the scalability of the model proposed by Lanillos & Cheng [8] as Gaussian processes regression is a crucial part of their algorithm which may prove to scale poorly. Specifically, through the introduction of sparse Gaussian process regression. If this approach works, this would allow the predictive processing model to be applied to more complex systems, including humanoid robots.

Chapter 2

Research

2.1 Methods

2.1.1 Mathematical model

Perception model

The mathematical model as proposed by Lanillos & Cheng [8] is based on works from predictive processing [6] and free energy approaches to perception [3, 2, 8]. The robot is defined as a set of sensors s and an unobserved variable $\hat{\mathbf{x}}$ that represents the most plausible hypothesis of the body variables (or joint angles in this case). The proprioceptive sensor $\mathbf{s}_p$ represents the body configuration and follows a multivariate normal distribution with a linear or non-linear mean $h_p(\mathbf{x})$ and covariance $\mathbf{\Sigma}_p$, this means $p(\mathbf{s}_p|\mathbf{x}) = \mathcal{N}(\mathbf{x}|g_p(\mathbf{x}), \mathbf{\Sigma}_p)$. The visual sensor $\mathbf{s}_v$ represents the location of the end-effector in the visual field and also follows a Normal distribution with a linear or non-linear mean $g_v(\mathbf{x})$ and covariance Σ_v, which means $p(\mathbf{s}_v|\mathbf{x}) = \mathcal{N}(\mathbf{x}|g_v(\mathbf{x}), \mathbf{\Sigma}_v)$.

To infer the body state, or joint configurations, visual and proprioceptive sensory information can be combined using Bayes rule. Assuming that the proprioceptive and visual sensory signals are independent given the latent state $\mathbf{x}$, we can define the distribution of $\mathbf{x}$ as follows:

$$p(\mathbf{x}|\mathbf{s}_p, \mathbf{s}_v) = \frac{p(\mathbf{s}_p|\mathbf{x})p(\mathbf{s}_v|\mathbf{x})p(\mathbf{x})}{p(\mathbf{s}_p, \mathbf{s}_v)} \tag{2.1}$$

This formulation doesn't take into account the temporal nature of the robotic system, as joint angles can change over time. To take this into consideration I assume the following state-space model and generative function [13]:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{x}_{t-1} + \mathbf{q}_{t-1} \\ \mathbf{y}_t &= \mathbf{h}(\mathbf{x}_t) + \mathbf{r}_t \end{aligned} \tag{2.2}$$

Where $\mathbf{x}_t \in \mathbb{R}^M$ represents the body variables, or joint angles, $\mathbf{y}_t \in \mathbb{R}^{M+2}$ represents the measurements from both the proprioceptive sensor $\mathbf{q}_{t-1} \sim \mathbb{N}(0, \mathbf{Q})$ is the measurement noise, $\mathbf{r}_t \sim \mathbb{N}(0, \mathbf{R}_t)$, $s_p \in \mathbb{R}^M$ and visual sensor $s_v \in \mathbb{R}^2$, and $\mathbf{h}(\cdot) : \mathbb{R}^M \to \mathbb{R}^{M+2}$ is the measurement model function. Also note that the notation $\mathbf{x}_{n|m}$ represents the estimate of $\mathbf{x}$ at time n given observations up to and including at time $m \leq n$.

Using the functions in defined equation 2.2 and applying Bayes rule the posterior distribution of $\mathbf{x}_t$ can be written as follows:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{1}{\mathbf{Z}_t} p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) \tag{2.3}$$

Where $\mathbf{Z}_t$ is the normalizing constant. However, in the case that the observation model is non-linear the normalizing constant is difficult to compute ($p(\mathbf{x}_t|\mathbf{y}_{1:t-1}$ can also be difficult to compute). To deal with this issue we need to make use of approximations. Lanillos & Cheng's model achieves this by minimizing the Kullback-Leibler divergence through the free-energy with lower bound F [8]. This approach, as noted by Lanillos & Cheng, is quite similar to an Extended Kalman filter (EKF) which is a commonly used alternative in engineering. Because I'm not concerned about biological plausibility, as I want to assess the scalability of the Gaussian Process regression for it's application within robotics, I have opted to implement an EKF for the predictive state estimation instead. The EKF utilizes a Taylor series approximation to linearize the system[13].

Predictive state estimation

The EKF assumes Gaussian approximations to the marginal posterior distribution of the state $\mathbf{x}_t$ at each time step t given the history of the measurements up to the time step t, which results in the following system of equations:

- Prediction:

$$\begin{aligned} \hat{\mathbf{x}}_{t|t-1} &= \hat{\mathbf{x}}_{t-1|t-1} \\ \mathbf{P}_{t|t-1} &= \mathbf{P}_{t-1|t-1} + \mathbf{Q} \end{aligned} \tag{2.4}$$

- Update:

$$\begin{aligned} \mathbf{v}_t &= \mathbf{y}_t - \mathbf{h}(\hat{\mathbf{x}}_{t|t-1}) \\ \mathbf{S}_t &= \mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R} \\ \mathbf{K}_t &= \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1} \\ \hat{\mathbf{x}}_{t|t} &= \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \mathbf{v}_t \\ \mathbf{P}_{t|t} &= (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1} \end{aligned} \tag{2.5}$$

Where $\mathbf{P}_{t|t-1} \in \mathbb{R}^{M \times M}$ is the predicted covariance, $\mathbf{v}_t \in \mathbb{R}^{M+2}$ represents the sensor error terms, $\mathbf{S}_t \in \mathbb{R}^{M+2 \times M+2}$ represents the sum of the predicted

covariance and the sensor noise covariance, $\mathbf{H}_t \in \mathbb{R}^{M+2\times M}$ is the Jacobian matrix of $\mathbf{h}(\hat{\mathbf{x}}_{t|t-1})$.

Body learning

To use the EKF for predictive state estimation we need to learn the forward model $\mathbf{y}_t = \mathbf{h}(\mathbf{x}_t)+\mathbf{r}_t$ and compute its derivative $\mathbf{H}(\mathbf{x}_t)$. They can be learned using a Gaussian Process regression model applied to the joint configurations $\mathbf{x}_t$ and the sensor values $\mathbf{y}_t$. To train the model I generated data by taking sensor samples $\mathbf{y}_t$ while the robot cycles through several body configurations $\mathbf{X} \in \mathbb{R}^{N\times M}$, see section 2.1.2.

I used this data to compute the covariance matrix $\mathbf{K}(\mathbf{X}, \mathbf{X})$ where the covariance function $\mathbf{k}(x_i, x_j)$ is defined as follows:

$$\mathbf{k}_{ij} = \mathbf{k}(x_i, x_j) + \sigma_n^2 \delta_{ij} \quad \text{where } x_i, x_j \in \mathbf{X} \tag{2.6}$$

Note that δ_{ij} is a Kronecker delta which is one iff $i = j$ and zero otherwise [10]. This allows us to define the predicted sensory outcome given our prediction from the previous time step as follows:

$$\mathbf{h}(\hat{\mathbf{x}}_{t|t-1}) = \mathbf{k}(\hat{\mathbf{x}}_{t|t-1}, \mathbf{X})\mathbf{K}(\mathbf{X}, \mathbf{X})^{-1}\mathbf{y}_t = \mathbf{k}(\hat{\mathbf{x}}_{t|t-1}, \mathbf{X})\mathbf{A} \tag{2.7}$$

Where we define $\mathbf{A} = \mathbf{K}(\mathbf{X}, \mathbf{X})^{-1}\mathbf{y}_t$, which, as it is not dependent on $\hat{\mathbf{x}}_{t|t-1}$, this can be precomputed for efficiency. Note that Lanillos and Cheng have directly encoded the internal belief in the proprioceptive space, this means for the first M dimensions $\mathbf{h}(\hat{\mathbf{x}}_{t|t-1}) = \hat{\mathbf{x}}_{t|t-1}$ and only the last 2 dimensions follow equation 2.7.

To compute the gradient of the sensory prediction $\mathbf{H}_t$ we differentiate the kernel as derived by McHutchon [10]. In line with the model proposed by Lanillos and Cheng I used the squared exponential covariance function, which gives me the following derivative:

$$\begin{aligned} \mathbf{H}_t(\hat{\mathbf{x}}_{t|t-1}) &= \frac{\partial \mathbf{h}(\hat{\mathbf{x}}_{t|t-1})}{\partial \hat{\mathbf{x}}_{t|t-1}} = \frac{\partial \mathbf{k}(\hat{\mathbf{x}}_{t|t-1}, \mathbf{X})}{\partial \hat{\mathbf{x}}_{t|t-1}}\mathbf{A} \\ &= -\mathbf{\Lambda}^{-1}\mathbf{Z}^T\mathbf{B} \end{aligned} \tag{2.8}$$

Here $\mathbf{\Lambda}$ is a matrix diagonally populated with the length-scale for each dimension d in $\hat{\mathbf{x}}_{t|t-1}$ $(\text{diag}(\boldsymbol{\ell}_d))$, $\mathbf{Z}$ is defined as $\mathbf{Z}_i = \hat{\mathbf{x}}_{t|t-1} - \mathbf{X}_i$ with $i \in \{1 \dots N\}$ where N is the number of samples in $\mathbf{X}$, and $\mathbf{B}$ is defined as $\mathbf{B}_i = (\mathbf{k}(\hat{\mathbf{x}}_{t|t-1}, \mathbf{X})^T \odot \mathbf{A}_i)$ with $i \in \{x, y\}$ representing the visual coordinates in the image and $\odot$ representing element-wise multiplication. Note again that because the internal belief is directly encoded in the proprioceptive space that the Jacobian for the proprioceptive sensor equal is to 1, thus we only need to compute the Jacobian for the visual sensor according to equation 2.8.

However, as noted before, Gaussian Processes are known not the scale well computationally. This is why I chose to also implement Sparse Gaussian Process regression (SGP) which is a common sparse approach to scale standard Gaussian Proccess regression (GP). In the SGP approach the data set is reduced to a subset of induced variables $\mathbf{u}$ which can be seen as summarizing $\mathbf{X}$. $\mathbf{u}$ is a vector that contains values of $\hat{\mathbf{x}}$ at point $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^{I}$ which live in the same space as $\mathbf{X}$. Note, that this means that $\mathbf{u}$ needs to contain values that are in the same space as $\mathbf{X}$, but they don't necessarily need to be values that are in $\mathbf{X}$. To improve effieciency we define $Z < N$, this makes the size covariance matrix smaller and therefor more efficient to invert. I used $I = 25$ randomly selected data points from $\mathbf{X}$ to initialize the inducing variables.

To relate the inducing point to the rest of the data set we fit a variational distribution $\mathbf{q}(\mathbf{u})$ [9]:

$$
\begin{aligned}
&q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{\Lambda}) \\
&\text{where:} \\
&\quad \mathbf{\Lambda} = \mathbf{K}(\mathbf{u},\mathbf{u})^{-1} + \mathbf{K}(\mathbf{u},\mathbf{u})^{-1}\mathbf{k}(\mathbf{u},\mathbf{X})\mathbf{k}(\mathbf{X},\mathbf{u})\mathbf{K}(\mathbf{u},\mathbf{u})^{-1}\sigma^{-2} \\
&\quad \mathbf{m} = \mathbf{\Lambda}^{-1}\mathbf{K}(\mathbf{u},\mathbf{u})^{-1}\mathbf{k}(\mathbf{u},\mathbf{X})\mathbf{y}_t\sigma^{-2}
\end{aligned}
\tag{2.9}
$$

I used the same data as for the standard GP to compute the covariance matrix $\mathbf{K}(\mathbf{u},\mathbf{u})$ which is the covariance function evaluated between all the inducing points and the covariance function $\mathbf{k}(x_i, x_j)$ follows the same equation as defined in 2.6, where $x_i, x_j \in \mathbf{u}$ as I used the same kernel. This allows us to define the predicted sensory outcome as follows:

$$
\mathbf{h}(\hat{\mathbf{x}}_{t|t-1}) = \mathbf{k}(\hat{\mathbf{x}}_{t|t-1}, \mathbf{u})\mathbf{K}(\mathbf{u},\mathbf{u})^{-1}\mathbf{m} \tag{2.10}
$$

Notice that if we define $\mathbf{A} = \mathbf{K}(\mathbf{u},\mathbf{u})^{-1}\mathbf{m}$ it becomes very similar to the prediction function defined for the standard GP in equation 2.7, the only difference being that the covariance function is evaluated between the estimated $\hat{\mathbf{x}}_{t|t-1}$ and the inducing points instead of the whole dataset $\mathbf{X}$. This mean that if we can find a good fit on the inducing variables our Jacobian should be similar to the one defined in equation 2.8. To test this I have plotted the numerical gradient and the analytical gradient computed with equation 2.8 against each other for both the GP and the SGP in figures 2.1 and 2.2 respectively.

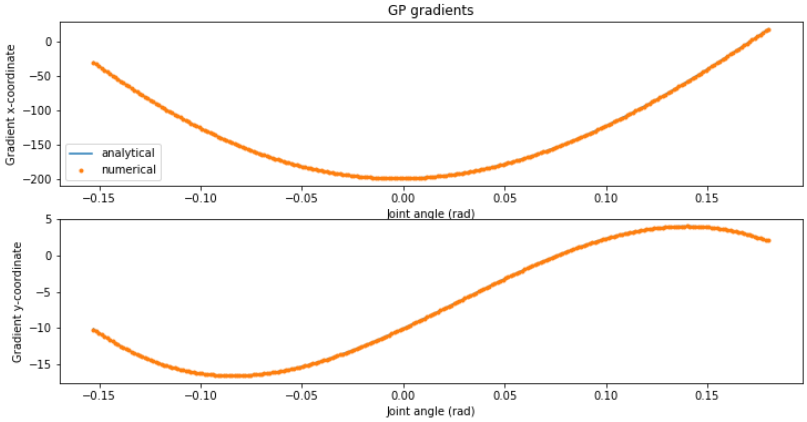

Figure 2.1: Analytical and numerical gradient for the GP model

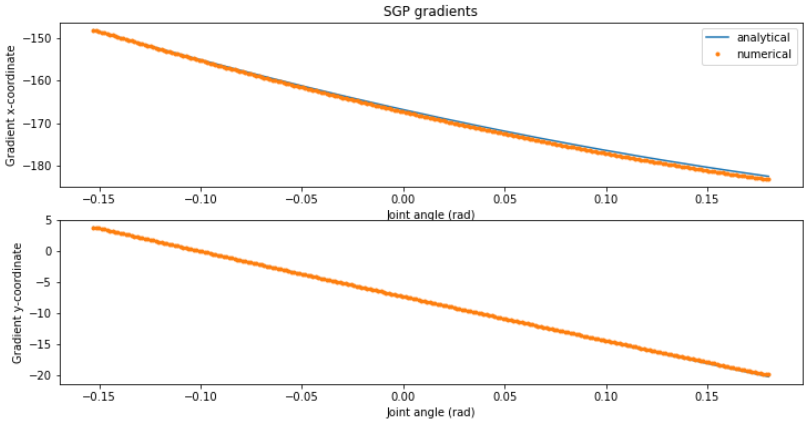

Figure 2.2: Analytical and numerical gradient for the SGP model

2.1.2 Simulations

Robotic setup

I used Webots (version: R2020b revision 1)[14] to simulate the robots, more specifically I used the UR5e, wich is a model that is build after the physical UR5e by Universal robots through sponsoring of the ROSin European project [4]. I made the following modifications to the robot. First I created a solid with a box shape of size $0.15.15 \times 1$m to function as the base, within this base is a camera which is positioned 0.20m from the top of the base slightly tilted down such that it has the entire arm in view. The UR5e is positioned 0.25m from the ground, rotated 90° to attach to the side of the

base. I also attached a spherical shape to the end-effector of the robot which I made visible to the camera so I can track the end-effector, figure 2.3 shows the robotic setup in Webots.

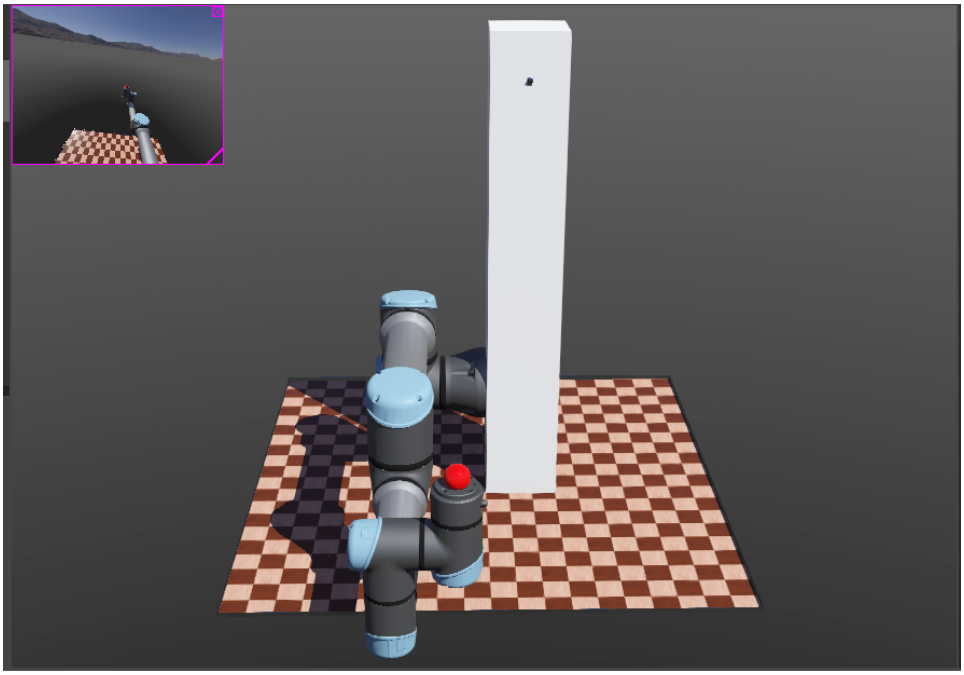

Figure 2.3: Robotic setup

Because I want assess the scalability of the algorithm I used the same robotic setup, varying in the number of joints $M \in \{1, 3, 6\}$ I use. To generate the training data I had the robot move through some trajectories mostly in the horizontal plane with added noise that follows a von Mises distribution with $\mu = 0$ and $\kappa = 2000$. All trajectories consist of $N = 250$ samples. The training trajectory for $M = 1$ consists of the shoulder 2 joint which moves the arm in the horizontal plane, 20 cm in one way and then 40 cm the other way. The trajectory for $M = 3$ is consist of the shoulder 1 joint, the shoulder 2 joint and the elbow joint. The shoulder 1 joint moves the arm in the vertical plane 2,5 cm up and then 5 cm down, shoulder 2 follows the same trajectory as for $M = 1$, and the elbow joint moves 5 cm one way and 10 cm the other way. For $M = 6$ the training and testing trajectories are extended with three wrist joints. The shoulder joints and the elbow joints follow the same trajectories as for $M = 3$. The first wrist joint moves 3,5 cm ones way and 5 cm the other way, the other two wrist joints move 1 cm one way and 2 cm the other way. The test trajectory for all numbers of joints is the flipped training trajectory (i.e. the end of the training trajectory is the start of the test trajectory). While the robot was cycling through the trajectories we saved the values of the proprioceptive

and visual sensors.

2.2 Results

As my goals was to asses the scalability of the GP and SGP models, I measured the accuracy of the training, the estimation error and the computational efficiency of both models over three different joint levels

2.2.1 Single joint

In figure 2.4 I plotted the training data for each visual coordinate (black crosses) along with the prediction obtained from the GP model. We can see from this plot that the GP has found a reasonable fit to the data. The plot also makes it clear that we assume the same amount of noise for each visual coordinate, something which could be improved upon.

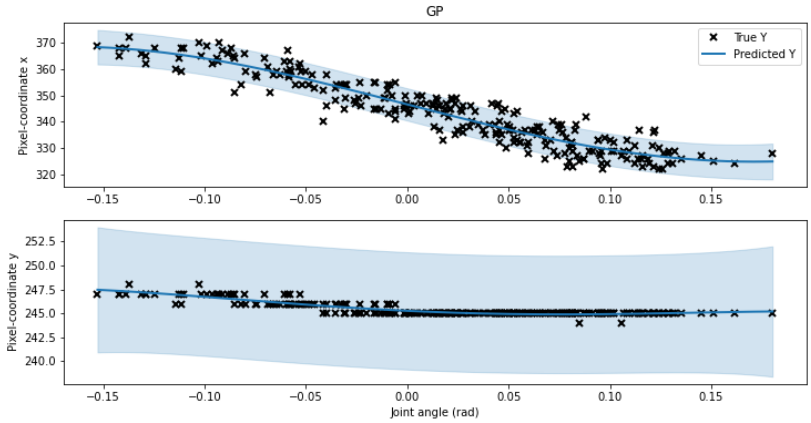

Figure 2.4: GP fit to the data

In figure 2.5 I again plotted the training data for each visual coordinate along with the prediction obtained from the SGP model. We can see in this plot that the SGP has also found a reasonable fit to the data, and again that the visual coordinates do not have the same amount of noise.

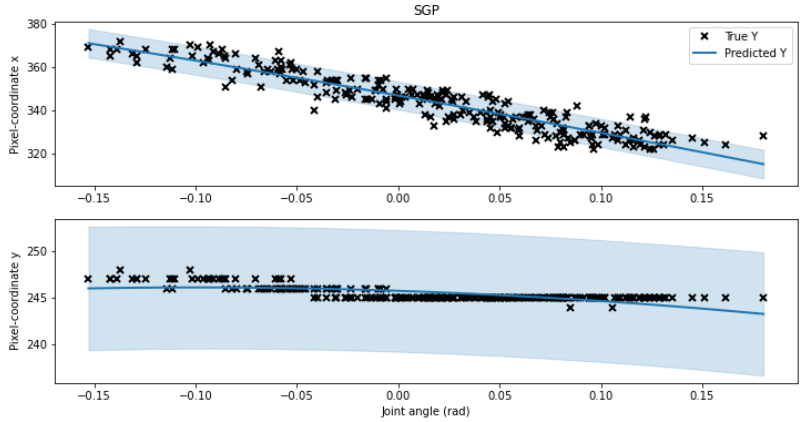

Figure 2.5: SGP fit to the data

In figure 2.6 we can see the body estimation by both the GP and SGP. Both models seem to have a decent predictive state estimate of the joint angle, although it is interesting to note that the standard GP actually seems to perform slightly worse than the SGP. This is possibly because the fitted GP, which we can see in figure 2.4 is more curved than the fitted SGP (figure 2.5) which means the gradient at those points is also more curved. This also shows in figures 2.1 and 2.2 which show the respective gradients for the GP and the SGP.

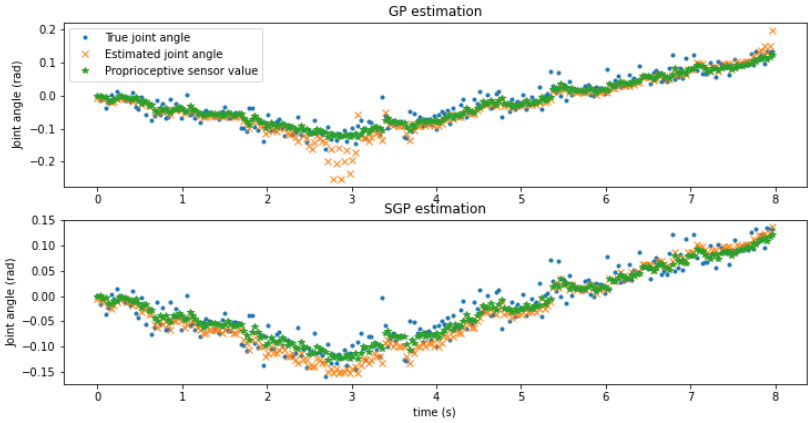

Figure 2.6: Estimation of $\hat{\mathbf{x}}_t$

2.2.2 Three joints

In the three joint case we have the problem that we can no longer show the fitted model to the data as done in figure 2.4 and 2.5 as our data now contains three dimensions, one for each joint angle. Figure 2.7 shows the predicted visual coordinate plotted against the true visual coordinate over all joint angles where the orange line represents the perfect prediction. We can see that both models have a similar fit to the data.

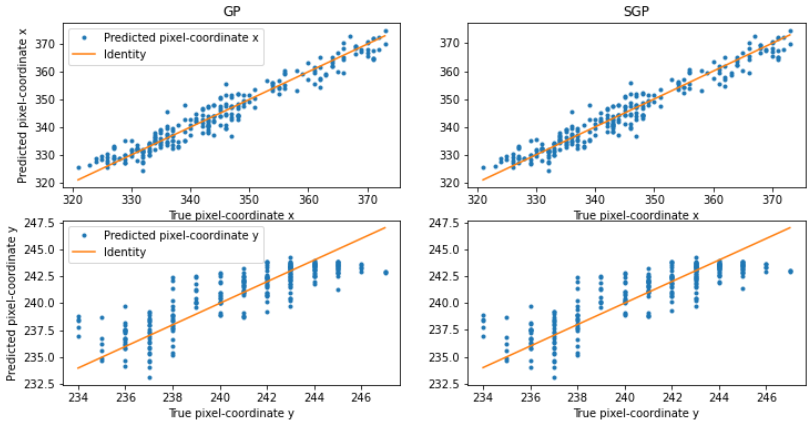

Figure 2.7: Model fit to the data

In figure 2.8 we can see the body estimation by the GP. The model seems to have difficulty with estimating the joins properly which is likely due to the fact that with three joint angles there is not necessarily always a unique solution to the system.

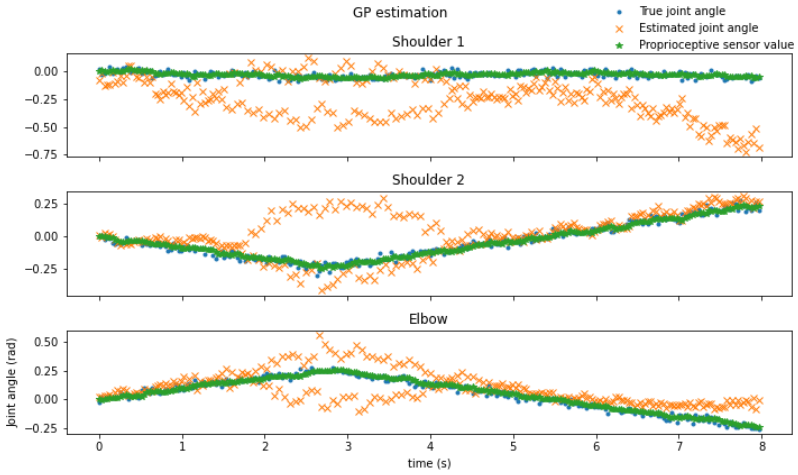

Figure 2.8: Estimation of $\hat{\mathbf{x}}_t$ of the GP

In figure 2.9 we can see the body estimation by the SGP. This model also seems to have difficulty with estimating the joins properly which is not surprising considering the fit of the standard GP. Interesting is that the SGP seems to estimate the shoulder 1 joint better than the GP, but seems to perform worse on the estimation of the shoulder 2 and elbow joints.

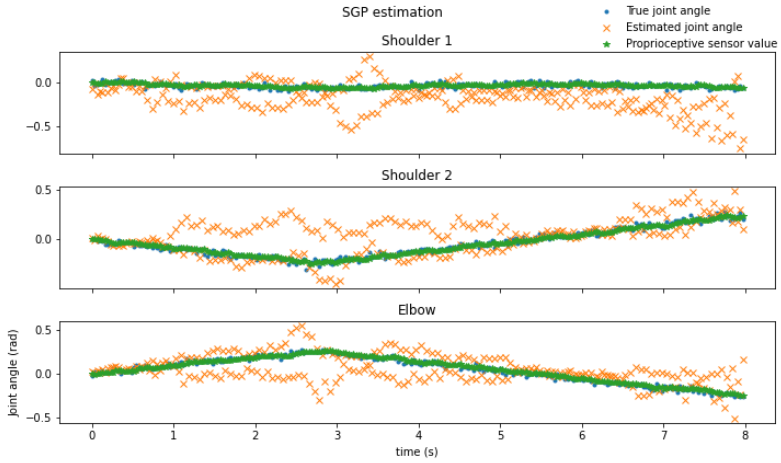

Figure 2.9: Estimation of $\hat{\mathbf{x}}_t$ of the SGP

2.2.3 Six joints

Figure 2.10 shows the predicted visual coordinate plotted against the true visual coordinate over all joint angles where the orange line represents the perfect prediction. We can see that again both models seem to have a similar fit tot the data.

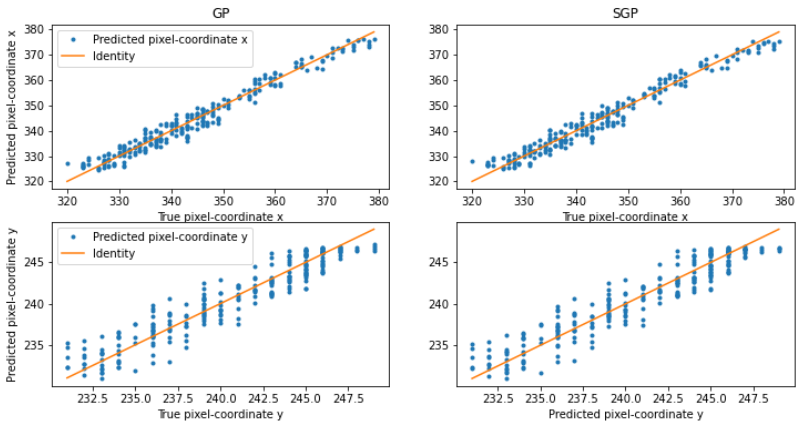

Figure 2.10: Model fit to the data

In figure 2.11 we can see the body estimation by the GP. The model seems to have even more difficulty with estimating the joins properly which is likely due to the fact that we introduced three more joints, which makes it even harder for the model to find a solution to the system.

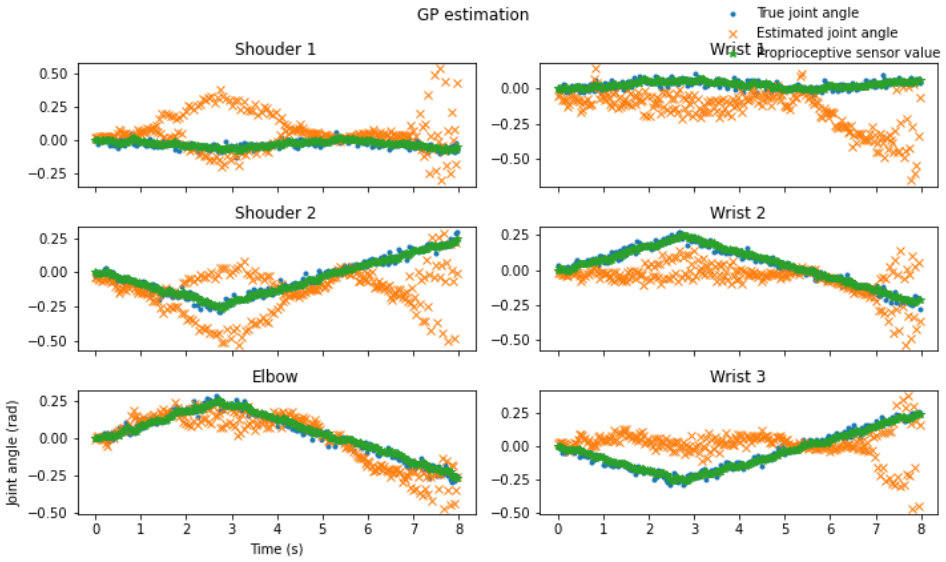

Figure 2.11: Estimation of $\hat{\mathbf{x}}_t$ for the GP

In figure 2.12 shows the body estimation by the SGP. We can see that there is not really a significant difference between the two models

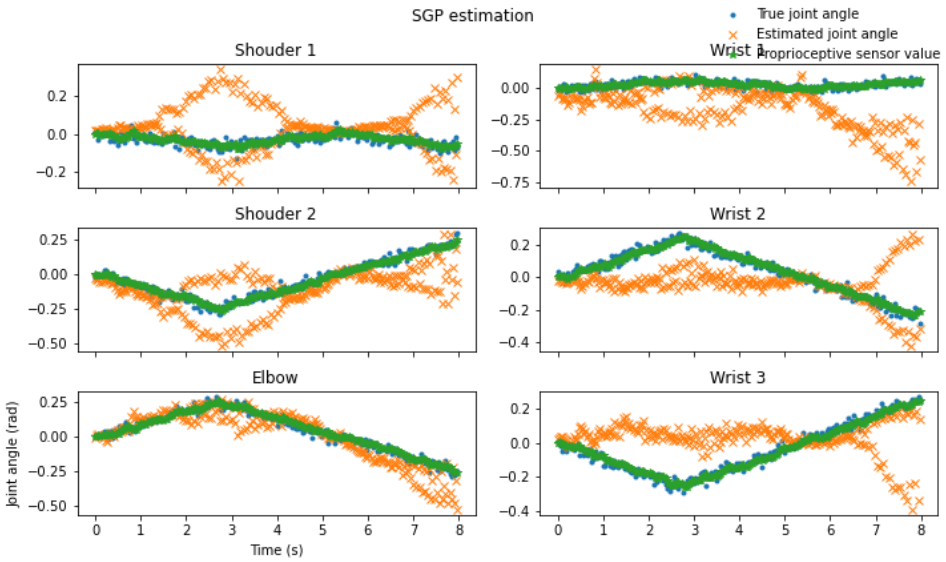

Figure 2.12: Estimation of $\hat{\mathbf{x}}_t$ for the SGP

Table 2.1 shows the average time the model took to predict the visual coordinates from $\hat{\mathbf{x}}_{t|t-1}$ in seconds. As expected the SGP is faster than the GP because the inverse of the covariance matrix is a smaller computation as we only use the inducing points $I = 25$ instead of all the samples $N = 250$.

	$m = 1$	$m = 3$	$m = 6$
GP	0.0028	0.0029	0.0030
SGP	0.0011	0.0011	0.0014

Table 2.1: Average time in seconds it takes to predict the visual coordinates

Chapter 3

Conclusions & Future directions

While, I found that the SGP model is faster in generating predictions, I have not encountered any issues with the standard GP which would warrant the use of an SGP. It is important to note that I only used data sets consisting 250 data points, which for a data set is quite small. A better way to test this would be to use bigger data sets, which might make the SGP model more valuable, as the GP model may not be able to handle the data size. Getting more data over a wider set of trajectories might also prove useful for the state estimation, as we can slightly see in figure 2.6 there are some difficulties in the estimation around the edges of the training data (e.g. 0.1 and -0,1), which might be because of the limited range of the trajectories, which may also explain some of the bad performance for the 3-joint and 6-joint systems. This issue may be mitigated by this wider range of trajectories. However, another reason for this difficulty in estimation might also be the fact that I am mapping 3 or 6 joint angles to only 2 visual coordinates, which in some cases may not have a unique solution. This could be improved by adding more camera's or other additional sensors, which may also make the SGP more useful as it would scale better to this increase in dimensionality.

Another future direction would be to improve the estimation of the sensory noise used in the estimation procedure, for simplicity I used fixed values for R and Q within the estimation, however, it would be good to include the proprioceptive sensor into the measurement function $\mathbf{h}(\cdot)$ and use the GP to estimate the noise terms for each dimension as well as $\mathbf{h}(\cdot)$. Doing this would allow for different amounts of noise between different proprioceptive sensors and visual coordinates as well as allow the model to account for potential sensor miscalibration or damage, as the GP can learn the offset of the sensor.

Another consideration is that, even though the SGP does not seem beneficial for the robotic systems I used, they might be for more complex systems

with more degrees of freedom more humanoid robots. For example, Atlas comprises 28 degrees of freedom [1] which may be impossible to compute with a GP, making an SGP more valuable.

Bibliography

[1] *Atlas*, `https://www.bostondynamics.com/atlas`, Accessed: 29-01-2021.

[2] Rafal Bogacz, *A tutorial on the free-energy framework for modelling perception and learning*, Journal of mathematical psychology **76** (2017), 198–211.

[3] Christopher L Buckley, Chang Sub Kim, Simon McGregor, and Anil K Seth, *The free energy principle for action and perception: A mathematical review*, Journal of Mathematical Psychology **81** (2017), 55–79.

[4] Alexander Ferrein, Stefan Schiffer, and Stephan Kallweit, *The rosin education concept*, Iberian Robotics conference, Springer, 2017, pp. 370–381.

[5] Karl Friston, *A theory of cortical responses*, Philosophical transactions of the Royal Society B: Biological sciences **360** (2005), no. 1456, 815–836.

[6] ———, *Hierarchical models in the brain*, PLoS Comput Biol **4** (2008), no. 11, e1000211.

[7] James Hensman, Nicolo Fusi, and Neil D Lawrence, *Gaussian processes for big data*, arXiv preprint arXiv:1309.6835 (2013).

[8] Pablo Lanillos and Gordon Cheng, *Adaptive robot body learning and estimation through predictive coding*, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 4083–4090.

[9] Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke. Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman, *GPflow: A Gaussian process library using TensorFlow*, Journal of Machine Learning Research **18** (2017), no. 40, 1–6.

[10] Andrew McHutchon, *Differentiating gaussian processes*, Cambridge (ed.) (2013).

[11] Rajesh PN Rao and Dana H Ballard, *Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects*, Nature neuroscience **2** (1999), no. 1, 79–87.

[12] Carl Edward Rasmussen and Christopher KI Williams, *Gaussian processes for machine learning. number isbn 0-262-18253-x*, 2006.

[13] Simo Särkkä, *Bayesian filtering and smoothing*, no. 3, Cambridge University Press, 2013.

[14] Webots, *http://www.cyberbotics.com*, Commercial Mobile Robot Simulation Software.