The Near Constant Acceleration Gaussian Process Kernel for Tracking

Steven Reece and Stephen Roberts

***Abstract*—Time series prediction is traditionally the domain of the state-based Kalman filter and very general Kalman filter process models, such as the near constant acceleration model (NCAM), have been developed to successfully track moving targets. However, the standard Kalman filter uses Markov process models and, consequently, it is difficult to track processes which include a complex periodic component. Gaussian processes are a generalisation of the Kalman filter and are able to model periodic behaviour efficiently and succinctly. However, no equivalent Gaussian process model for near constant acceleration has been formulated. We develop an equivalent Gaussian process kernel for NCAM to be used for time-series prediction.**

***Index Terms*—Bayesian methods, Gaussian processes, Kalman filter, near constant acceleration model, periodic dynamics, target tracking.**

I. Introduction

GAUSSIAN processes (GP) are experiencing a resurgence of interest. Current applications are in diverse fields such as geophysics [10], crowd tracking [2], sensor placement [3] and time series prediction [6]. Time series prediction is traditionally the domain of the state-based Kalman filter and very general Kalman filter process models, such as the near constant acceleration model (NCAM), have been developed to successfully track moving targets. However, the standard Kalman filter uses Markov process models and, consequently, it is difficult to track processes which include a complex periodic component [9]. Gaussian processes, on the other hand, are able to model periodic behaviour efficiently and succinctly. Thus, we develop an equivalent Gaussian process kernel for NCAM to be used for time-series prediction.

The letter is structured as follows. Section II gives a brief description of the Gaussian process and kernels which underpin them. Section III then presents the kernel for the near constant acceleration model. The efficacy of this kernel is demonstrated via a target tracking application in Section IV. We conclude in Section V.

Manuscript received March 26, 2010; revised May 05, 2010; accepted May 18, 2010. Date of publication June 07, 2010; date of current version June 21, 2010. This work was performed as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Networks) project which was jointly supported by a BAE Systems and EPSRC strategic partnership (EP/C548051/1) and was also funded by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Markku Renfors.

The authors are with the Robotics Research Group, Department of Engineering Science, Oxford University, Oxford, U.K. (e-mail: reece@robots.ox.ac.uk).

Digital Object Identifier 10.1109/LSP.2010.2051620

II. Gaussian Processes

A GP is often thought of as a Gaussian distribution over functions [7]. A GP is fully described by its mean and covariance. Suppose we have a set of training data $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ drawn from a noisy process:

$$y_i = f(x_i) + \epsilon_i \tag{1}$$

where ϵ_i is a zero-mean Gaussian random variable with variance σ^2. For convenience both inputs and outputs are aggregated into sets $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$, respectively. The GP estimates the value of the function f at test inputs $X_* = \{x_{*1}, \ldots, x_{*m}\}$. The basic GP regression equations are given in [7]:

$$\bar{f}_* = K(X_*, X)[K(X, X) + \sigma^2 I]^{-1} Y, \tag{2}$$

$$\mathrm{Cov}(f_*) = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma^2 I]^{-1} K(X_*, X)^T \tag{3}$$

where I is the identity matrix, $\bar{f}_*$ is the posterior mean function at X_* and $\mathrm{Cov}(f_*)$ is the posterior covariance. The matrix $K(X, X)$ is the covariance of the Gaussian prior distribution over the function f. The covariance matrix has elements:

$$K(x_i, x_j) = \mathrm{Cov}(f(x_i), f(x_j)) \, .$$

The term $K(X_*, X)$ is the covariance between the test inputs X_* and the training inputs X. The function K is alternatively called the *kernel* or the *covariance function*.

The GP parameters θ (which includes σ and parameters associated with the covariance function) are the hyperparameters of the GP which can be inferred through Bayes' rule:

$$Pr(\theta \mid Y, X) = \frac{Pr(Y \mid X, \theta)}{Pr(Y \mid X)} Pr(\theta) \, .$$

The hyperparameters are usually given a vague prior distribution $Pr(\theta)$.

III. The Near Constant Acceleration Model

The near constant acceleration model (NCAM) was developed in the 1970's [4], chiefly as a model for target tracking within the Kalman filter [1]. The NCAM is a general expression for all second order differentiable functions f which are subject to a random diffusion ϵ:

$$f(x_k) = f(x_{k-1}) + (x_k - x_{k-1}) f'(x_{k-1}) + \frac{1}{2}(x_k - x_{k-1})^2 f''(x_{k-1}) + \epsilon_k$$

where the time variable, x_k, increases in value with increasing index k (i.e., $x_i > x_j$ if and only if $i > j$).

Defining the *state vector* ϕ_k at time x_k to be

$$\phi_k \triangleq \begin{pmatrix} f(x_k) \\ f'(x_k) \\ f''(x_k) \end{pmatrix}$$

and the temporal difference $\delta_k = x_k - x_{k-1} > 0$ then

$$\phi_k = F_k \phi_{k-1} + q\epsilon_k, \tag{4}$$

$$\bar{\phi}_k = F_k \bar{\phi}_{k-1} \tag{5}$$

where $\bar{\phi}$ is the mean of the Gaussian random variable ϕ and

$$F_k = F(\delta_k) = \begin{pmatrix} 1 & \delta_k & \frac{\delta_k^2}{2} \\ 0 & 1 & \delta_k \\ 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

The scalar $q \geq 0$ is the acceleration coefficient and ϵ is assumed to be a zero-mean Gaussian random variable with covariance [4]:

$$Q_k = Q(\delta_k) = \begin{pmatrix} \frac{\delta_k^5}{20} & \frac{\delta_k^4}{8} & \frac{\delta_k^3}{6} \\ \frac{\delta_k^4}{8} & \frac{\delta_k^3}{3} & \frac{\delta_k^2}{2} \\ \frac{\delta_k^3}{6} & \frac{\delta_k^2}{2} & \delta_k \end{pmatrix}.$$

Note that, by choosing δ sufficiently large, the NCAM can determine the mean and covariance for the process f for *any* future time. Also note that

$$F(\delta_i)F(\delta_j) = F(\delta_i + \delta_j) \tag{7}$$

and

$$F(\delta_i)Q(\delta_j)F(\delta_i)^T + Q(\delta_i) = Q(\delta_i + \delta_j)\,. \tag{8}$$

We now derive a covariance function equivalent to the NCAM which is a necessary step to implementing the NCAM in a Gaussian process. From (4) and (5):

$$\begin{aligned} \mathrm{Cov}(\phi_i, \phi_i) &= F_i \mathrm{Cov}(\phi_{i-1}, \phi_{i-1}) F_i^T + qQ_i\,, \\ \mathrm{Cov}(\phi_i, \phi_j) &= F_i \mathrm{Cov}(\phi_{i-1}, \phi_j) \qquad (i > j). \end{aligned}$$

Using (7) and (8) we can simplify these expressions and remove the need for recursion:

$$\begin{aligned} \mathrm{Cov}(\phi_i, \phi_i) &= F(x_i - x_0)\mathrm{Cov}(\phi_0, \phi_0)F(x_i - x_0)^T \\ &\quad + qQ(x_i - x_0) \\ \mathrm{Cov}(\phi_i, \phi_j) &= F(x_i - x_j)\mathrm{Cov}(\phi_j, \phi_j) \qquad (i > j) \\ &= F(x_i - x_j)[F(x_j - x_0)\mathrm{Cov}(\phi_0, \phi_0)F(x_j - x_0)^T \\ &\quad + qQ(x_j - x_0)] \\ &= F(x_i - x_0)\mathrm{Cov}(\phi_0, \phi_0)F(x_j - x_0)^T \\ &\quad + qF(x_i - x_j)Q(x_j - x_0). \end{aligned}$$

Thus, defining the following vectors:

$$\begin{aligned} M(\delta) &\triangleq \begin{pmatrix} 1 & \delta & \frac{\delta^2}{2} \end{pmatrix}, \\ N(\delta) &\triangleq \begin{pmatrix} \frac{\delta^5}{20} & \frac{\delta^4}{8} & \frac{\delta^3}{6} \end{pmatrix} \end{aligned}$$

and, recalling that the time indices are ordered so that $i > j \iff x_i > x_j$, then the NCAM GP covariance function $K_{ij}^{\mathrm{NCAM}} = \mathrm{Cov}(f(x_i), f(x_j))$ is

$$\begin{aligned} K_{ij}^{\mathrm{NCAM}} &= M(x_i - x_0)\,\mathrm{Cov}(\phi_0, \phi_0)M(x_j - x_0)^T \\ &\quad + qM(x_i - x_j)N(x_j - x_0)^T \quad (x_i \geq x_j). \end{aligned}$$

This is the general expression for the nonstationary NCAM kernel and it has eight hyperparameters: six independent values in $\mathrm{Cov}(\phi_0, \phi_0)$, the "origin" x_0 and the acceleration coefficient q. The reader is warned that the expansion of this expression can be ugly.

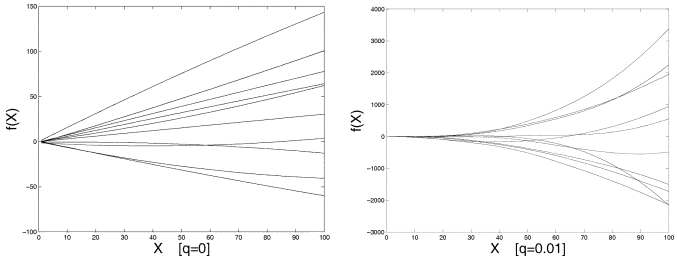

Fig. 1. Typical functions randomly generated from an NCAM GP prior.

The NCAM GP kernel makes clear the relationship between the Kalman filter parameters, especially q, in (4) and the identical Gaussian process hyperparameters (including q). As a matter of fact, the nonparametric GP model is equivalent to the Kalman filter parametric model in this case. We note that, although ϕ is a Markov process, the subprocess f by itself is not Markovian and this allows a smooth curve to be fitted to the data points [8]. Fig. 1 shows sample one-dimensional functions generated by the NCAM kernel for various values of the scaling parameter q.

The NCAM GP kernel was developed for a one dimensional signal. It is straightforward to extend the kernel to higher dimensional problems when the signals are a priori uncorrelated. In this case the prior covariance is block diagonal with one block for each dimension and each block calculated using K^{NCAM}.

IV. Application

We demonstrate the efficacy of the NCAM GP on a one dimensional target tracking problem that would be difficult to implement using a Kalman filter [9]. The target moves according to NCAM (with $\mathrm{Cov}(\phi_0, \phi_0) = \mathbf{0}$ and the acceleration coefficient set to $q = 10^{-5}$) and is tracked by a range sensor. This sensor is mounted on a platform which, is itself, subject to a cyclic perturbation in its elevation (see Fig. 3). For example, the sensor could be mounted on a small boat which rises and falls with the waves. We assume that the perturbation function is not known a priori and that it comprises multiple modes so that it is not simply sinusoidal.

Fig. 2 shows a typical data stream for a tracked target which moves according to NCAM and is sensed using an elevation sensor which rises and falls periodically. This data is generated using a target trajectory sampled from the NCAM GP prior and platform dynamics sampled from the following squared-exponential periodic kernel [5], [7]:

$$K^{\mathrm{PER}}(x, x') = \mu_{\mathrm{SEP}} \exp\left(-\frac{2\sin^2\left(\frac{x - x'}{2}\right)}{l_{\mathrm{SEP}}^2} \right) \tag{9}$$

where the input and output scale hyperparameters are chosen to be $l_{\mathrm{SEP}} = 10$ and $\mu_{\mathrm{SEP}} = 50$, respectively. The target is sensed every five time steps.

When $f(x)$ and $p(x)$ are the target's location and the platform's elevation at time x, respectively, then the sensor's measurement $y(x)$ is

$$y(x) = f(x) - p(x) + \epsilon_x \tag{10}$$

where ϵ_x is a zero-mean Gaussian random variable with variance σ^2. In our experiments $\sigma^2 = 5$. We note that the measurement model in (10) differs from the basic GP measurement

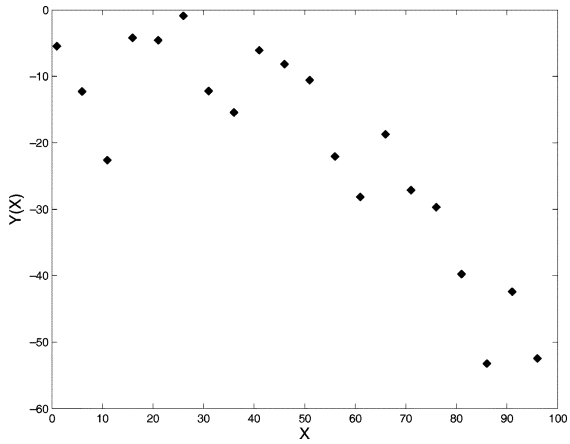

Fig. 2. Sensor data for target following an NCAM trajectory obtained from a platform undergoing periodic motion.

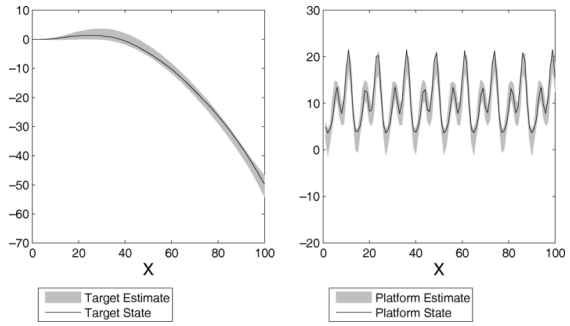

Fig. 3. Target location and platform elevation inferred using NCAM GP from the data in Fig. 2. Also shown is the ground truth.

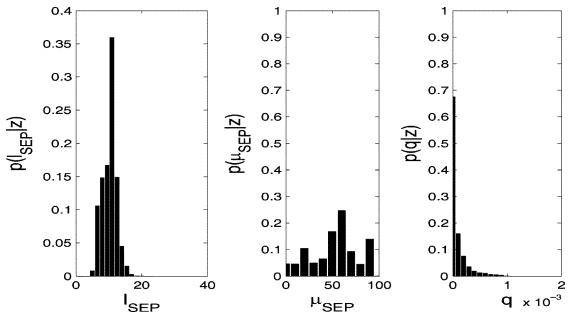

Fig. 4. Marginal hyperparameter posterior probabilities. These are the sensor platform's input scale l_{SEP}, output scale μ_{SEP} and the target's acceleration coefficient q.

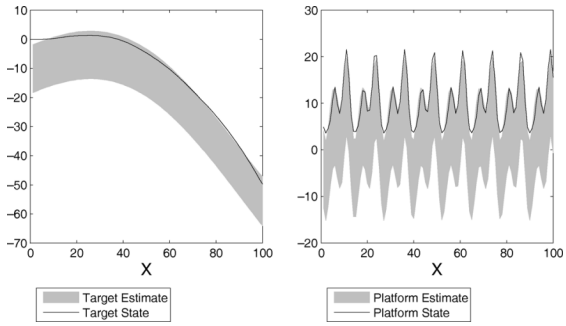

Fig. 5. Target's location and platform's elevation inferred from data in Fig. 2. Also shown is the ground truth. This example uses a squared-exponential target trajectory model (SE) in place of the more appropriate NCAM model.

model in (1) and the functions of interest, namely the target's trajectory and the platform dynamics, are hidden variables. Consequently, the GP equations, (2) and (3), are adapted in order to determine the means, $\bar{f}$ and $\bar{p}$, and covariances, $\mathrm{Cov}(f)$ and $\mathrm{Cov}(p)$, of the posterior distributions of the target trajectory and platform dynamics:

$$\begin{aligned}\bar{f}(X_*) &= K^{\mathrm{NCAM}}(X_*, X)\Sigma^{-1}\, Y \\ \mathrm{Cov}(f(X_*)) &= K^{\mathrm{NCAM}}(X_*, X_*) - K^{\mathrm{NCAM}}(X_*, X) \\ &\quad \times \Sigma^{-1} K^{\mathrm{NCAM}}(X_*, X)^T \\ \bar{p}(X_*) &= -K^{\mathrm{PER}}(X_*, X)\Sigma^{-1}\, Y\,, \\ \mathrm{Cov}(p(X_*)) &= K^{\mathrm{PER}}(X_*, X_*) - K^{\mathrm{PER}}(X_*, X) \\ &\quad \times \Sigma^{-1} K^{\mathrm{PER}}(X_*, X)^T\end{aligned}$$

where

$$\Sigma = K^{\mathrm{NCAM}}(X, X) + K^{\mathrm{PER}}(X, X) + \sigma^2 I\,.$$

The inputs X are the times at which the target is observed, Y are the observations and X_* are the times for which the target and platform positions are inferred. K^{PER} is the squared exponential periodic kernel in (9).

Fig. 3 shows the first standard deviation for the target's location and the platform's elevation posterior distributions inferred using the NCAM GP and the data in Fig. 2. Also shown is the ground truth for comparison. Fig. 4 shows the marginal hyperparameter posterior probabilities. The GP successfully infers both the target's location and the platform's elevation everywhere.

The hyperparameters $\theta = \{q, \mu_{\mathrm{SEP}}, l_{\mathrm{SEP}}\}$ were sampled directly from their priors. We assumed a priori that q, μ_{SEP} and l_{SEP} were independent and uniformly distributed within the closed intervals $[0, 10^{-3}]$, [0, 100] and [1, 20], respectively. We note that the matrices corresponding to $M(x_i - x_0)\ \mathrm{Cov}(\phi_0, \phi_0)M(x_j - x_0)^T$ and $M(x_i - x_j)N(x_j - x_0)^T$ within the expression for K^{NCAM} are independent of q and need not be recalculated for each new sample q. This leads to an efficient sample based implementation of the GP. In our implementation we use 1000 $\{q, \mu_{\mathrm{SEP}}, l_{\mathrm{SEP}}\}$ samples to compute the results in Fig. 3.

To demonstrate the efficacy of the NCAM kernel over existing GP kernels, we replace the NCAM kernel with the popular squared exponential (SE) kernel [7]:

$$K^{\mathrm{SE}}(x, x') = \mu_{\mathrm{SE}} \exp\left(-\frac{(x - x')^2}{l_{\mathrm{SE}}^2}\right)$$

and re-infer the platform and target trajectories from the data in Fig. 2. This time, the GP determines four stationary hyperparameters which are the periodic squared-exponential kernel input and output scales, l_{SEP} and μ_{SEP} respectively, and the input and output scales, l_{SE} and μ_{SE} respectively, of the target trajectory squared-exponential kernel. The results are shown in Fig. 5. The GP is less successful in this case and exhibits considerably greater uncertainty in the target's location and the sensor's platform dynamics. To quantify the efficacy of the NCAM and SE models we collected data from 100 runs where the target dynamics and cyclic platform dynamics were randomly sampled each time. The RMSE for the platform

TABLE I
MEAN RMSE FOR NCAM, SE AND SEL WITH TARGET ACCELERATION COEFFICIENT $q = 10^{-5}$

Model	Mean RMSE Platform	Target
NCAM	2.34	1.20
SE	8.45	8.24
SEL	2.80	1.97

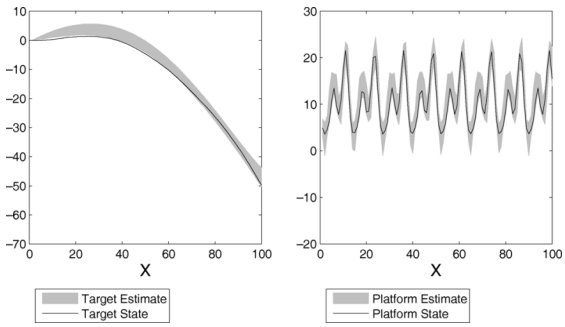

Fig. 6. Target's location and platform's elevation inferred from data in Fig. 2. Also shown is the ground truth. This example uses a product linear and squared-exponential target trajectory model (SEL) in place of the more appropriate NCAM model.

TABLE II
MEAN RMSE FOR NCAM AND SEL FOR THREE TARGET ACCELERATION COEFFICIENTS q

q	Model	Mean RMSE Platform	Target
0.01	NCAM	3.20	2.70
	SEL	4.38	3.95
0.1	NCAM	3.92	3.95
	SEL	7.29	6.31
1	NCAM	5.47	6.18
	SEL	13.37	13.21

location and target dynamics for these runs, averaged over time, are shown in the first two rows in Table I.

We can improve the performance of the SE model by combining it with a linear kernel which mimics the increasing variance of the NCAM over time:

$$K^{\mathrm{SEL}}(x, x') = \mu_{\mathrm{SEL}} x x' \exp\left(-\frac{(x - x')^2}{l^2_{\mathrm{SEL}}}\right).$$

The results for this kernel are shown in Fig. 6 and it is evident that the SEL kernel offers a good approximation to the NCAM kernel in this case. The RMSE for the 100 runs, averaged over time, for the SEL model is presented in the third row of Table I.

Table II compares the NCAM and SEL models for larger acceleration coefficients (i.e., $q \in \{0.01, 0.1, 1\}$). Example results for $q = 1$ are shown in Fig. 7. It is evident from this example and the table that the SEL ceases to be a good approximation to the NCAM for larger values of q. However, we must emphasise that the poor showing of the off-the-shelf GP solutions is largely due to the experimental design which strongly favours the NCAM-GP solution.

V. CONCLUSIONS

This letter presents, for the first time, a Gaussian process covariance function for the familiar state-based near constant acceleration model which is a fundamental component of Kalman

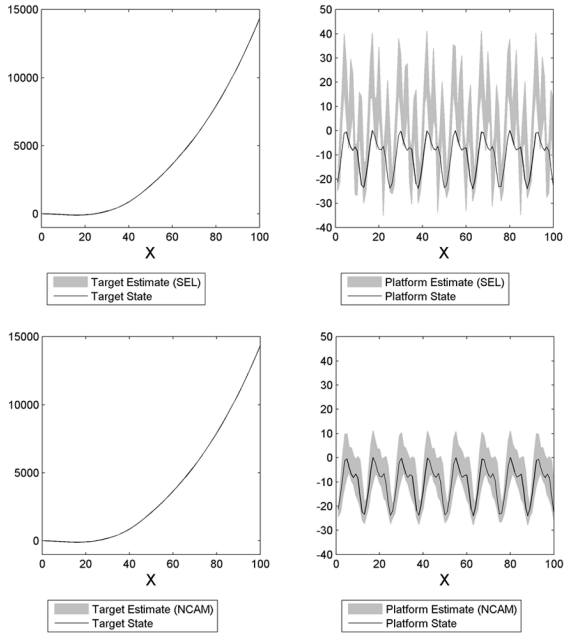

Fig. 7. Target's location and platform's elevation. Also shown is the ground truth. This example compares the NCAM (bottom graphs) against the SEL target trajectory model (top graphs) for a target with acceleration coefficient $q = 1$.

filters in tracking applications. The efficacy of this kernel was demonstrated on a target tracking problem for which the target is tracked by a sensor which undergoes unknown periodic motion. This problem is difficult to solve using a Kalman filter for arbitrary and unknown periodic sensor dynamics. The Gaussian process was able to infer both the target trajectory and the platform dynamics. Although the NCAM kernel can be approximated by existing GP kernels these may not capture the full range of NCAM dynamics. Thus, we believe that the NCAM kernel is an important addition to the GP kernel repertoire.

REFERENCES

[1] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. New York: Academic, 1988.
[2] D. Ellis, E. Sommerlade, and I. Reid, "Modelling pedestrian trajectory patterns with Gaussian processes," in *Proc. 11th IEEE Int. Conf. Embedded and Real-Time Computing Systems and Applications (RTCSA '05)*, 2005, pp. 1533–2306.
[3] C. Guestrin, A. Krause, and A. Singh, "Near-optimal sensor placements in Gaussian processes," in *22nd Int. Conf. Machine Learning (ICML)*, 2005, ICML, pp. 265–272.
[4] A. H. Jaswinski, *Stochastic Processes and Filtering Theory*. New York: Academic, 1970.
[5] D. J. C. Mackay, "Introduction to Gaussian processes," in *Neural Netw. Mach. Learn.*, C. M. Bishop, Ed. Berlin, Germany: Springer-Verlag, 1998.
[6] M. Osborne, A. Rogers, A. Ramchurn, S. Roberts, and N. R. Jennings, "Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes," in *IPSN 2008: International Conference on Information Processing in Sensor Networks*, St. Louis, MO, 2008.
[7] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.
[8] S. Reece and S. Roberts, "An introduction to Gaussian processes for the Kalman filter expert," in *Proc. 13th IEEE Int. Conf. Information Fusion (FUSION '10)*, Edinburgh, U.K., 2010.
[9] S. Särkkä, A. Vehtari, and J. Lampinen, "Prediction of estsp competition time series by unscented Kalman filter and rts smoother," in *Proc. Eur. Symp. Time Series Prediction (ESTSP)*, Espoo, Finland, 2007.
[10] S. Vasudevan, F. Ramos, E. Nettleton, and H. Durrant-Whyte, "Gaussian process modeling of large scale terrain," in *Proc. Ninth IEEE Int. Workshop on Visual Surveillance*, Kyoto, Japan, 2009.