CLMC Technical Report Number: TR-CLMC-2007-1

Learning an Outlier-Robust Kalman Filter

Jo-Anne Ting, Evangelos Theodorou, Stefan Schaal
{joanneti, etheodor, sschaal} @ usc.edu

Computational Learning & Motor Control Laboratory
University of Southern California

June 28, 2007

Learning an Outlier-Robust Kalman Filter

Jo-Anne Ting[1], Evangelos Theodorou[1] and Stefan Schaal[1,2]

[1] University of Southern California, Los Angeles, CA 90089
[2] ATR Computational Neuroscience Laboratories, Kyoto, Japan
{joanneti, etheodor, sschaal}@usc.edu

Abstract. In this paper, we introduce a modified Kalman filter that performs robust, real-time outlier detection, without the need for manual parameter tuning by the user. Systems that rely on high quality sensory data (for instance, robotic systems) can be sensitive to data containing outliers. The standard Kalman filter is not robust to outliers, and other variations of the Kalman filter have been proposed to overcome this issue. However, these methods may require manual parameter tuning, use of heuristics or complicated parameter estimation procedures. Our Kalman filter uses a weighted least squares-like approach by introducing weights for each data sample. A data sample with a smaller weight has a weaker contribution when estimating the current time step's state. Using an incremental variational Expectation-Maximization framework, we learn the weights and system dynamics. We evaluate our Kalman filter algorithm on synthetic data and data from a robotic dog.

1 Introduction

Systems that rely on high quality sensory data are often sensitive to data containing outliers. While data from sensors such as potentiometers and optical encoders are easily interpretable in their noise characteristics, other sensors such as visual systems, GPS devices and sonar sensors often provide measurements populated with outliers. As a result, robust, reliable detection and removal of outliers is essential in order to process these kinds of data. For example, in the application domain of robotics, legged locomotion is vulnerable to sensory data of poor quality, since one undetected outlier can disturb the balance controller to the point that the robot loses stability.

An outlier is generally defined as an observation that "lies outside some overall pattern of distribution" [1]. Outliers may originate from sensor noise (producing values that fall outside a valid range), from temporary sensor failures, or from unanticipated disturbances in the environment (e.g., a brief change of lighting conditions for a visual sensor). Note that some prior knowledge about the observed data's properties must be known. Otherwise, it is impossible to discern if a data sample that lies some distance away from the data cloud is truly an outlier or simply part of the data's structure.

For real-time applications, storing data samples may not be a viable option due to the high frequency of sensory data and insufficient memory resources.

In this scenario, sensor data are made available one at a time and must be discarded once they have been observed. Hence, techniques that require access to the entire set of data samples, such as the Kalman smoother (e.g., [2,3]), are not applicable. Instead, the Kalman filter [4] is a more suitable method, since it assumes that only data samples up to the current time step have been observed.

The Kalman filter is a widely used tool for estimating the state of a dynamic system, given noisy measurement data. It is the optimal *linear* estimator for linear Gaussian systems, giving the minimum mean squared error [5]. Using state estimates, the filter can also estimate what the corresponding (output) data are. However, the performance of the Kalman filter degrades when the observed data contains outliers. To address this, previous work has tried to make the Kalman filter more robust to outliers by addressing the sensitivity of the squared error criterion to outliers [6,7]. One class of approaches considers non-Gaussian distributions for random variables (e.g., [8–11]), since multivariate Gaussian distributions are known to be susceptible to outliers. For example, [12] uses multivariate Student-t distributions. However, the resulting estimation of parameters may be quite complicated for systems with transient disturbances.

Alternatively, it is possible to model the observation and state noise as non-Gaussian, heavy-tailed distributions to account for non-Gaussian noise and outliers (e.g., [13–15]). Unfortunately, these filters are typically more difficult to implement and may no longer provide the conditional mean of the state vector. Other approaches use resampling techniques (e.g., [16,17]) or numerical integration (e.g., [18,19]), but these may require heavy computation not suitable for real-time applications.

Yet another class of methods uses a weighted least squares approach, as done in robust least squares [20], where the measurement residual error is assigned some statistical property. Some of these algorithms fall under the first category of approaches as well, assuming non-Gaussian distributions for variables. Each data sample is assigned a weight that indicates its contribution to the hidden state estimate at each time step. This technique has been used to produce a Kalman filter that is more robust to outliers (e.g., [21,22]). However, these methods usually model the weights as some heuristic function of the data (e.g., the Huber function [20]) and often require manual tuning of threshold parameters for optimal performance. Using incorrect or inaccurate estimates for the weights may lead to deteriorated performance, so special attention and care is necessary when using these techniques.

In this paper, we are interested in making the Kalman filter more robust to the outliers in the observations (i.e. the filter should identify and eliminate possible outliers as it tracks observed data). Identifying outliers in the state is an entirely different problem, left for another paper. We introduce a modified Kalman filter that can detect outliers in the observed data without the need for manual parameter tuning or use of heuristic methods. This filter learns the weights of each data sample and the system dynamics, using an incremental Expectation-Maximization (EM) framework [23]. For ease of analytical computation, we assume Gaussian distributions for variables and states. We illustrate the performance of this robust Kalman filter on synthetic and robotic data, comparing it with other robust Kalman filter methods and demonstrating its effectiveness at detecting outliers in the observations.

2 Outlier Detection in the Kalman Filter

Let us assume we have data observed over N time steps, $\{\mathbf{z}_k\}_{k=1}^N$, and the corresponding hidden states as $\{\boldsymbol{\theta}_k\}_{k=1}^N$ (where $\boldsymbol{\theta}_k \in \Re^{d_2\times 1}, \mathbf{z}_k \in \Re^{d_1\times 1}$). Assuming a time-invariant system, the Kalman filter system equations are:

$$\begin{aligned} \mathbf{z}_k &= \mathbf{C}\boldsymbol{\theta}_k + \mathbf{v}_k \\ \boldsymbol{\theta}_k &= \mathbf{A}\boldsymbol{\theta}_{k-1} + \mathbf{s}_k \end{aligned} \tag{1}$$

where $\mathbf{C} \in \Re^{d_1\times d_2}$ is the observation matrix, $\mathbf{A} \in \Re^{d_2\times d_2}$ is the state transition matrix, $\mathbf{v}_k \in \Re^{d_1\times 1}$ is the observation noise at time step k, and $\mathbf{s}_k \in \Re^{d_2\times 1}$ is the state noise at time step k. We assume $\mathbf{v}_k$ and $\mathbf{s}_k$ to be uncorrelated additive mean-zero Gaussian noise: $\mathbf{v}_k \sim \text{Normal}\,(0, \mathbf{R})$, $\mathbf{s}_k \sim \text{Normal}\,(0, \mathbf{Q})$, where $\mathbf{R} \in \Re^{d_1\times d_1}$ is a diagonal matrix with $\mathbf{r} \in \Re^{d_1\times 1}$ on its diagonal, and $\mathbf{Q} \in \Re^{d_2\times d_2}$ is a diagonal matrix with $\mathbf{q} \in \Re^{d_2\times 1}$ on its diagonal. $\mathbf{R}$ and $\mathbf{Q}$ are covariance matrices for the observation and state noise, respectively. Fig. 1(a) shows the graphical model for the standard Kalman filter. Its corresponding filter propagation and update equations are, for $k = 1, .., N$:

Propagation:

$$\boldsymbol{\theta}'_k = \mathbf{A}\left\langle \boldsymbol{\theta}_{k-1}\right\rangle \tag{2}$$

$$\boldsymbol{\Sigma}'_k = \mathbf{A}\boldsymbol{\Sigma}_{k-1}\mathbf{A}^T + \mathbf{Q} \tag{3}$$

Update:

$$\mathbf{S}'_k = \left(\mathbf{C}\boldsymbol{\Sigma}'_k\mathbf{C}^T + \mathbf{R}\right)^{-1} \tag{4}$$

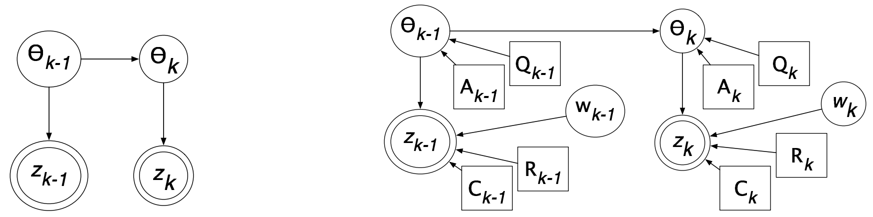
$$K'_k = \boldsymbol{\Sigma}'_k\mathbf{C}^T\mathbf{S}'_k \tag{5}$$

$$\left\langle \boldsymbol{\theta}_k\right\rangle = \boldsymbol{\theta}'_k + K'_k\left(\mathbf{z}_k - \mathbf{C}\boldsymbol{\theta}'_k\right) \tag{6}$$

$$\boldsymbol{\Sigma}_k = \left(\mathbf{I} - K'_k\mathbf{C}\right)\boldsymbol{\Sigma}'_k \tag{7}$$

where $\langle\boldsymbol{\theta}_k\rangle$[3] is the posterior mean vector of the state $\boldsymbol{\theta}_k$, $\boldsymbol{\Sigma}_k$ is the posterior covariance matrix of $\boldsymbol{\theta}_k$, and $\mathbf{S}'_k$ is the covariance matrix of the residual prediction error—all at time step k. In this problem, the system dynamics ($\mathbf{C}$, $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{Q}$) are unknown, and it is possible to use a maximum likelihood framework to estimate these parameter values [24]. Unfortunately, this standard Kalman filter model considers all data samples to be part of the data cloud and is not robust to outliers.

[3] Note that $\langle\rangle$ denotes the expectation operator



(a) Kalman filter (b) Robust Kalman filter with Bayesian weights

Fig. 1. Graphical Models: circular nodes are random variables, double circles are observed random variables, and square nodes are point estimated parameters.

2.1 Robust Kalman Filtering with Bayesian Weights

To overcome this limitation, we introduce a novel Bayesian algorithm that treats the weights associated with each data sample probabilistically. In particular, we introduce a scalar weight w_k for each observed data sample $\mathbf{z}_k$ such that the variance of $\mathbf{z}_k$ is weighted with w_k, as done in [25]. [25] considers a weighted least squares regression model and assumes that the weights are known and given. We model the weights to be Gamma distributed random variables, as done previously in [26] for weighted linear regression. Additionally, we learn estimates for the system dynamics at each time step. A Gamma prior distribution is chosen for the weights in order to ensure they remain positive. Fig. 1(b) shows the graphical model of this robust Kalman filter. The resulting prior distributions are then:

$$\begin{aligned} \mathbf{z}_k|\boldsymbol{\theta}_k, w_k &\sim \text{Normal}\left(\mathbf{C}\boldsymbol{\theta}_k, \mathbf{R}/w_k\right) \\ \boldsymbol{\theta}_k|\boldsymbol{\theta}_{k-1} &\sim \text{Normal}\left(\mathbf{A}\boldsymbol{\theta}_{k-1}, \mathbf{Q}\right) \\ w_k &\sim \text{Gamma}\left(a_{w_k}, b_{w_k}\right) \end{aligned} \tag{8}$$

We can treat this entire problem as an Expectation-Minimization-like (EM) learning problem [23, 27] and maximize the log likelihood $\log p(\boldsymbol{\theta}_{1:N})$ (known as the "incomplete" log likelihood with the hidden probabilistic variables marginalized out). Due to analytical issues, we only have access to a lower bound of this measure. This lower bound is based on an expected value of the "complete" data likelihood $\langle \log p\left(\boldsymbol{\theta}_{1:N}, \mathbf{z}_{1:N}, \mathbf{w}\right)\rangle$, formulated over all variables of the learning problem:

$$\begin{aligned} &\log p\left(\boldsymbol{\theta}_{1:N}, \mathbf{z}_{1:N}, \mathbf{w}\right) \\ &= \textstyle\sum_{i=1}^{N} \log p\left(\mathbf{z}_i|\boldsymbol{\theta}_i, w_i\right) + \sum_{i=1}^{N} \log p\left(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}\right) + \log p\left(\boldsymbol{\theta}_0\right) + \sum_{i=1}^{N} \log p\left(w_i\right) \end{aligned} \tag{9}$$

where $\mathbf{w} \in \Re^{N\times 1}$ has coefficients w_i $(i = 1, .., N)$, and $\mathbf{z}_{1:N}$ denotes samples $\{\mathbf{z}_1, \mathbf{z}_2, .., \mathbf{z}_N\}$. Since we are considering this problem as a real-time one (i.e.

data samples arrive sequentially, one at a time), we will have observed only data samples $\mathbf{z}_{1:k}$ at time step k. Consequently, in order to estimate the posterior distributions of the random variables and parameter values at time step k, we should consider the log evidence of only the data samples observed to date, i.e., $\log p\left(\boldsymbol{\theta}_{1:k}, \mathbf{z}_{1:k}, \mathbf{w}_{1:k}\right)$.

The expectation of the complete data likelihood should be taken with respect to the true posterior distribution of all hidden variables $Q\left(\mathbf{w}, \boldsymbol{\theta}\right)$. Since this is an analytically intractable expression, we use a technique from variational calculus to construct a lower bound and make a factorial approximation of the true posterior as follows: $Q\left(\mathbf{w}, \boldsymbol{\theta}\right) = \prod_{i=1}^{N} Q\left(w_i\right) \prod_{i=1}^{N} Q\left(\boldsymbol{\theta}_i | \boldsymbol{\theta}_{i-1}\right) Q(\boldsymbol{\theta}_0)$ (e.g., [27]). This factorization of $\boldsymbol{\theta}$ considers the influence of each $\boldsymbol{\theta}_i$ from within its Markov blanket, conserving the Markov property that Kalman filters, by definition, have. While losing a small amount of accuracy, all resulting posterior distributions over hidden variables become analytically tractable. This factorial approximation was chosen purposely so that $Q(w_k)$ is independent from $Q(\boldsymbol{\theta}_k)$; performing joint inference of w_k and $\boldsymbol{\theta}_k$ does not make sense in the context of our generative model. The final EM update equations at time step k are:

E-step:

$$\boldsymbol{\Sigma}_k = \left(\langle w_k \rangle \mathbf{C}_k^T \mathbf{R}_k^{-1} \mathbf{C}_k + \mathbf{Q}_k^{-1}\right)^{-1} \tag{10}$$

$$\langle \boldsymbol{\theta}_k \rangle = \boldsymbol{\Sigma}_k \left(\mathbf{Q}_k^{-1} \mathbf{A}_k \langle \boldsymbol{\theta}_{k-1} \rangle + \langle w_k \rangle \mathbf{C}_k^T \mathbf{R}_k^{-1} \mathbf{z}_k\right) \tag{11}$$

$$\langle w_k \rangle = \frac{a_{w_k,0} + \frac{1}{2}}{b_{w_k,0} + \left\langle \left(\mathbf{z}_k - \mathbf{C}_k \boldsymbol{\theta}_k\right)^T \mathbf{R}_k^{-1} \left(\mathbf{z}_k - \mathbf{C}_k \boldsymbol{\theta}_k\right) \right\rangle} \tag{12}$$

M-step:

$$\mathbf{C}_k = \left(\sum_{i=1}^{k} \langle w_i \rangle \mathbf{z}_i \langle \boldsymbol{\theta}_i \rangle^T\right) \left(\sum_{i=1}^{k} \langle w_i \rangle \left\langle \boldsymbol{\theta}_i \boldsymbol{\theta}_i^T \right\rangle\right)^{-1} \tag{13}$$

$$\mathbf{A}_k = \left(\sum_{i=1}^{k} \langle \boldsymbol{\theta}_i \rangle \langle \boldsymbol{\theta}_{i-1} \rangle^T\right) \left(\sum_{i=1}^{k} \left\langle \boldsymbol{\theta}_{i-1} \boldsymbol{\theta}_{i-1}^T \right\rangle\right)^{-1} \tag{14}$$

$$r_{km} = \frac{1}{k} \sum_{i=1}^{k} \langle w_i \rangle \left\langle \left(\mathbf{z}_{im} - \mathbf{C}_k(m,:) \boldsymbol{\theta}_i\right)^2 \right\rangle \tag{15}$$

$$q_{kn} = \frac{1}{k} \sum_{i=1}^{k} \left\langle \left(\boldsymbol{\theta}_{in} - \mathbf{A}_k(n,:) \boldsymbol{\theta}_{i-1}\right)^2 \right\rangle \tag{16}$$

where $m = 1, .., d_1$, $n = 1, .., d_2$; r_{km} is the mth coefficient of the vector $\mathbf{r}_k$; q_{kn} is the nth coefficient of the vector $\mathbf{q}_k$; $\mathbf{C}_k(m,:)$ is the mth row of the matrix $\mathbf{C}_k$; $\mathbf{A}_k(n,:)$ is the nth row of the matrix $\mathbf{A}_k$; and $a_{w_k,0}$ and $b_{w_k,0}$ are prior scale parameters for the weight w_k. (10) to (16) should be computed once for each time step k (e.g., [28] [29]) when the data sample $\mathbf{z}_k$ becomes available.

Since storing sensor data is not possible in real-time applications, (13) to (16)—which require access to all observed data samples up to time step k—need to be re-written using only values observed, calculated or used in the current time step k. We can do this by collecting sufficient statistics in (13) to (16) and rewriting them as:

$$\mathbf{C}_k = \mathrm{sum}_k^{\mathbf{wz}\boldsymbol{\theta}^T} \left(\mathrm{sum}_k^{\mathbf{w}\boldsymbol{\theta}\boldsymbol{\theta}^T}\right)^{-1} \tag{17}$$

$$\mathbf{A}_k = \mathrm{sum}_k^{\boldsymbol{\theta}\boldsymbol{\theta}'} \left(\mathrm{sum}_k^{\boldsymbol{\theta}'\boldsymbol{\theta}'}\right)^{-1} \tag{18}$$

$$r_{km} = \frac{1}{k}\left[\mathrm{sum}_{km}^{\mathbf{w}zz} - 2\mathbf{C}_k(m,:)\mathrm{sum}_{km}^{\mathbf{w}z\boldsymbol{\theta}} + \mathrm{diag}\left\{\mathbf{C}_k(m,:)\mathrm{sum}_k^{\mathbf{w}\boldsymbol{\theta}\boldsymbol{\theta}^T}\mathbf{C}_k(m,:)^T\right\}\right] \tag{19}$$

$$q_{kn} = \frac{1}{k}\left[\mathrm{sum}_{kn}^{\theta^2} - 2\mathbf{A}_k(n,:)\mathrm{sum}_{kn}^{\theta\boldsymbol{\theta}'} + \mathrm{diag}\left\{\mathbf{A}_k(n,:)\mathrm{sum}_k^{\boldsymbol{\theta}'\boldsymbol{\theta}'}\mathbf{A}_k(n,:)^T\right\}\right] \tag{20}$$

where $m = 1, .., d_1$, $n = 1, .., d_2$, and the sufficient statistics, which are all a function of values observed, calculated or used in time step k (e.g., $\langle w_k \rangle$, $\mathbf{z}_k$, $\langle \boldsymbol{\theta}_k \rangle$, $\langle \boldsymbol{\theta}_{k-1} \rangle$ etc.) are:

$$\begin{aligned}
\mathrm{sum}_k^{\mathbf{wz}\boldsymbol{\theta}^T} &= \langle w_k \rangle \mathbf{z}_k \langle \boldsymbol{\theta}_k \rangle^T + \mathrm{sum}_{k-1}^{\mathbf{wz}\boldsymbol{\theta}^T} & \mathrm{sum}_k^{\mathbf{w}\boldsymbol{\theta}\boldsymbol{\theta}^T} &= \langle w_k \rangle \left\langle \boldsymbol{\theta}_k \boldsymbol{\theta}_k^T \right\rangle + \mathrm{sum}_{k-1}^{\mathbf{w}\boldsymbol{\theta}\boldsymbol{\theta}^T} \\
\mathrm{sum}_k^{\boldsymbol{\theta}\boldsymbol{\theta}'} &= \langle \boldsymbol{\theta}_k \rangle \langle \boldsymbol{\theta}_{k-1} \rangle^T + \mathrm{sum}_{k-1}^{\boldsymbol{\theta}\boldsymbol{\theta}'} & \mathrm{sum}_k^{\boldsymbol{\theta}'\boldsymbol{\theta}'} &= \left\langle \boldsymbol{\theta}_{k-1} \boldsymbol{\theta}_{k-1}^T \right\rangle + \mathrm{sum}_{k-1}^{\boldsymbol{\theta}'\boldsymbol{\theta}'} \\
\mathrm{sum}_{km}^{\mathbf{w}zz} &= \langle w_k \rangle z_{km}^2 + \mathrm{sum}_{k-1}^{\mathbf{w}zz} & \mathrm{sum}_{km}^{\mathbf{w}z\boldsymbol{\theta}} &= \langle w_k \rangle z_{km} \boldsymbol{\theta}_k + \mathrm{sum}_{k-1,m}^{\mathbf{w}z\boldsymbol{\theta}} \\
\mathrm{sum}_{kn}^{\theta^2} &= \left\langle \theta_{kn}^2 \right\rangle + \mathrm{sum}_{k-1,n}^{\theta^2} & \mathrm{sum}_{kn}^{\theta\boldsymbol{\theta}'} &= \langle \theta_{kn} \rangle \langle \boldsymbol{\theta}_{k-1} \rangle + \mathrm{sum}_{kn}^{\theta\boldsymbol{\theta}'}
\end{aligned}$$

A few remarks should be made regarding the initialization of priors used in (10) to (12), (17) to (20). In particular, the prior scale parameters $a_{w_k,0}$ and $b_{w_k,0}$ should be selected so that the weights $\langle w_k \rangle$ are 1 with some confidence. That is to say, the algorithm starts by assuming most data samples are inliers. For example, we set $a_{w_k,0} = 1$ and $b_{w_k,0} = 1$ so that $\langle w_k \rangle$ has a prior mean of $a_{w_k,0}/b_{w_k,0} = 1$ with a variance of $a_{w_k,0}/b_{w_k,0}^2 = 1$. By using these values, the maximum value of $\langle w_k \rangle$ is capped at 1.5. This set of values is generally valid for any data set and/or application and does not need to be modified if no prior information regarding the presence of outliers in the data is available. Otherwise, if the user has prior knowledge regarding the strong or weak presence of outliers in the data set (and hence, a good reason to insert strong biases towards particular parameter values), the prior scale parameters of the weights can be modified accordingly to reflect this. Since some prior knowledge about the observed data's properties must be known in order to distinguish whether a data sample is an outlier or part of the data's structure, this Bayesian approach provides a natural framework to incorporate this information.

Secondly, the algorithm is relatively insensitive to the the initialization of $\mathbf{A}$ and $\mathbf{C}$ and will always converge to the same final solution, regardless of these values. For our experiments, we initialize $\mathbf{C} = \mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Finally, the initial values of $\mathbf{R}$ and $\mathbf{Q}$ should be set based on the user's initial estimate of how noisy the observed data is (e.g., $\mathbf{R} = \mathbf{Q} = 0.01\mathbf{I}$ for noisy data, $\mathbf{R} = \mathbf{Q} = 10^{-4}\mathbf{I}$ for less noisy data [30]).

2.2 Relationship to the Kalman Filter

Equations (10) and (11) for the posterior mean and posterior covariance of $\boldsymbol{\theta}_k$ may not look like the standard Kalman filter equations in (2) to (7), but with a little algebraic manipulation, we can show that the model derived in Section 2.1 is indeed a variant of the Kalman filter. If we substitute the propagation equations, (2) and (3), into the update equations, (4) to (7), we reach recursive expressions for $\langle \boldsymbol{\theta}_k \rangle$ and $\boldsymbol{\Sigma}_k$. By applying this sequence of algebraic manipulations in reverse order to (10) and (11), we arrive at the following:

Propagation:

$$\boldsymbol{\theta}'_k = \mathbf{A}_k \langle \boldsymbol{\theta}_{k-1} \rangle \tag{21}$$

$$\boldsymbol{\Sigma}'_k = \mathbf{Q}_k \tag{22}$$

Update:

$$\mathbf{S}'_k = \left(\mathbf{C}_k \boldsymbol{\Sigma}'_k \mathbf{C}_k^T + \frac{1}{\langle w_k \rangle} \mathbf{R}_k \right)^{-1} \tag{23}$$

$$K'_k = \boldsymbol{\Sigma}'_k \mathbf{C}_k^T \mathbf{S}'_k \tag{24}$$

$$\langle \boldsymbol{\theta}_k \rangle = \boldsymbol{\theta}'_k + K'_k \left(\mathbf{z}_k - \mathbf{C}_k \boldsymbol{\theta}'_k \right) \tag{25}$$

$$\boldsymbol{\Sigma}_k = \left(\mathbf{I} - K'_k \mathbf{C}_k \right) \boldsymbol{\Sigma}'_k \tag{26}$$

Close examination of the above equations show that (10) and (11) in the Bayesian model correspond to standard Kalman filter equations, with modified expressions for $\boldsymbol{\Sigma}'_k$ and $\mathbf{S}'_k$ and time-varying system dynamics. $\boldsymbol{\Sigma}'_k$ is no longer *explicitly* dependent on $\boldsymbol{\Sigma}_{k-1}$ since $\boldsymbol{\Sigma}_{k-1}$ does not appear in (22). However, the current state's covariance $\boldsymbol{\Sigma}_k$ is still dependent on the previous state's covariance $\boldsymbol{\Sigma}_{k-1}$ (i.e. it is dependent through the other parameters K' and $\mathbf{C}_k$).

Additionally, the term $\mathbf{R}_k$ in $\mathbf{S}'_k$ is now weighted. Equation (12) reveals that if the prediction error in $\mathbf{z}_k$ is so large that it dominates the denominator, then the weight $\langle w_k \rangle$ of that data sample will be very small. As this prediction error term in the denominator goes to ∞, $\langle w_k \rangle$ approaches 0. If $\mathbf{z}_k$ has a very small weight $\langle w_k \rangle$, then $\mathbf{S}'_k$, the posterior covariance of the residual prediction error, will be very small, leading to a very small Kalman gain K'_k. In short, the influence of the data sample $\mathbf{z}_k$ will be downweighted when predicting $\boldsymbol{\theta}_k$, the hidden state at time step k.

The resulting Bayesian algorithm has a computational complexity on the same order as that of a standard Kalman filter, since matrix inversions are still needed (for the calculation of covariance matrices), as in the standard Kalman filter. In comparison to other Kalman filters that use heuristics or require more involved computation/implementation, this outlier-robust Kalman filter is principled and easy to implement.

2.3 Monitoring the Residual Error

A common sanity check is to monitor the residual error of the data $\mathbf{z}_{1:N}$ and the hidden states $\boldsymbol{\theta}_{1:N}$ in order to ensure that the residual error values stay within the 3σ bounds computed by the filter [30]. If we had access to the true state θ_k for time step k, we would plot the residual state error $(\boldsymbol{\theta}_k - \langle \boldsymbol{\theta}_k \rangle)$ for all time steps k, along with the corresponding $\pm 3\sigma_k$ values, where $\sigma_k^2 = \mathrm{diag}\{\boldsymbol{\Sigma}_k\}$. We would also plot the residual prediction error $(\mathbf{z}_k - \mathbf{CA}\langle \boldsymbol{\theta}_{k-1} \rangle)$ for all time steps k, along with the corresponding $\pm 3\sigma_{z_k}$ values, where $\sigma_{z_k}^2 = \mathrm{diag}\{\mathbf{S}'_k\}$.

With these graphs, we should observe the residual error values remaining within the $\pm 3\sigma$ bounds and check that the residual error does not diverge over time. Residual monitoring may be useful to verify that spurious data samples are rejected, since processing of these samples may result in corrupted filter computations. It offers a peek into the Kalman filter, providing insights as to how the filter performs.

2.4 An Alternative Kalman Filter

We explored a variation of the previously introduced robust Kalman filter. Instead of performing a full Bayesian treatment of the weighted Kalman filter, we use the standard Kalman filter equations, (2) to (7), and modify (4) in an ad hoc manner so that the output variance for $\mathbf{z}_k$, $\mathbf{R}_k$, is now weighted—as in our original model in (8):

$$\mathbf{S}'_k = \left(\mathbf{C}_k \boldsymbol{\Sigma}'_k \mathbf{C}_k^T + \frac{1}{\langle w_k \rangle} \mathbf{R}_k \right)^{-1} \tag{27}$$

We learn the weights $\langle w_k \rangle$ using (12) from the robust Kalman filter and estimate the system dynamics ($\mathbf{C}$, $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{Q}$) at each time step using a maximum likelihood framework (i.e., using (17) to (20) from the robust Kalman filter). In this alternative filter, $\boldsymbol{\Sigma}_k$ is *explicitly* dependent on $\boldsymbol{\Sigma}_{k-1}$ (i.e. $\boldsymbol{\Sigma}_{k-1}$ appears in the propagation equation for $\boldsymbol{\Sigma}_k$). We introduce this somewhat unprincipled and arbitrarily derived filter in order to compare it with our weighted Kalman filter. Details on the performances of both filters can be found in the next section.

3 Experimental Results

We evaluated our weighted robust Kalman filter on synthetic and robotic data sets and compared it with three other filters. We omitted the filters of [21] and [22], since we had difficulty implementing them and getting them to work. Instead, we used a hand-tuned thresholded Kalman filter to serve as a baseline comparison. The three filters consist of i) the standard Kalman filter, ii) the alternative weighted Kalman filter introduced in Section 2.4, and iii) a Kalman filter where outliers are determined by thresholding on the Mahalanobis distance. If the Mahalanobis distance is less than a certain threshold value, then it is considered an inlier and processed. Otherwise, it is an outlier and ignored. This threshold value is *hand-tuned manually* in order to find the optimal value for a particular data set. If we have a priori access to the entire data set and are able to tune this threshold value accordingly, the thresholded Kalman filter gives *near-optimal* performance.

First, we simulate a real data set where hidden states are unknown and only access to observed data is available. Although they are linear, Kalman filters are commonly used to track more interesting "nonlinear" behaviors (i.e., not just a straight line). For this reason, we try the methods on a synthetic data set exhibiting nonlinear behavior, where the system dynamics are unknown. We also conducted experiments on a synthetic data set where the system dynamics of the generative model are known. These experiments yield similar performance results to that where the system dynamics are unknown. Finally, we run all Kalman filters on data collected from a robotic dog, LittleDog, manufactured by Boston Dynamics Inc. (Cambridge, MA).

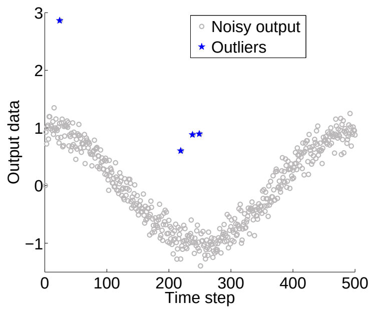
For this paper and these experiments, we are interested in the Kalman filter's prediction of the observed (output) data and detection of outliers in the observations. We are not interested in the estimation of the system dynamics or in the estimation (or outlier detection) of the states. Estimation of the system matrices for the purpose of parameter identification is a different problem and is not addressed in this paper. Details on this difference are highlighted in [31]. Similarly, detecting outliers in the states is a different problem and is left to another paper.

3.1 Synthetic Data with Unknown System Dynamics

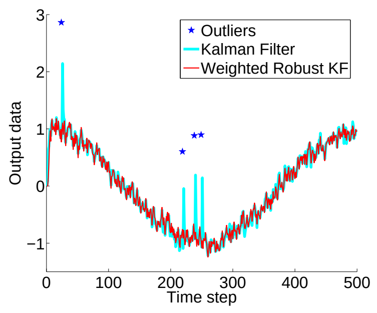
We created data exhibiting nonlinear behavior, where $\mathbf{C}$, $\mathbf{A}$, $\mathbf{R}$, $\mathbf{Q}$ and states are unknown, high noise is added to the (output) data, and a data sample is an outlier with 1% probability. One-dimensional data is used for ease of visualization, and Fig. 2(a) shows a noisy cosine function with outliers, over 500 time steps. For optimal performance, $\mathbf{C}$, $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{Q}$ were manually tuned for the standard Kalman filter—a tricky and time-consuming process. In contrast, the system dynamics were learnt for the thresholded Kalman filter using a maximum likelihood framework (i.e. using (17) to (20) without any weights).

Fig. 2(b) shows how sensitive the standard Kalman filter is to outliers, while the weighted robust Kalman filter seems to detect them quite well. In Fig. 2(c), we compare the weighted robust Kalman filter with the alternative filter and thresholded filter. All three filters appear to perform as well, which is unsurprising, given the amount of manual tuning required by the thresholded Kalman filter.

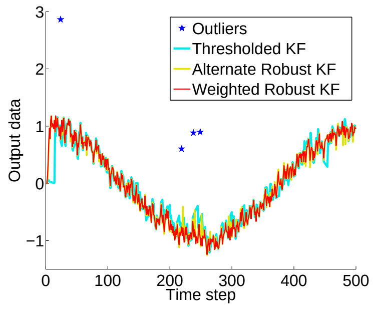
Fig. 2(d) shows that the residual prediction error on the outputs stays within the $\pm 3\sigma$ bounds. In Fig. 3, we can see that the covariance of the residual error is slightly smaller for the weighted robust filter (i.e. we are slightly more confident in our estimates for the weighted robust filter). This, in turn, translates to a slightly higher Kalman gain, K'_k, for the alternative filter (this is easily seen by plotting both Kalman gains). A higher K'_k means that more consideration is given to the sample $\mathbf{z}_k$ when estimating the current time step's hidden state. Graphs showing the estimated states were omitted due to lack of space, but they show similar trends in the accuracy results.



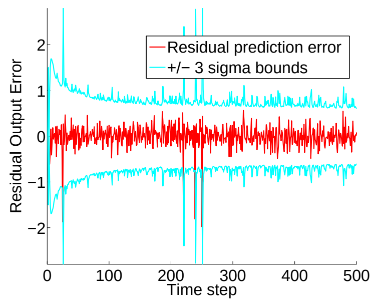
(a) Observed noisy output data with outliers



(b) Predicted data for the Kalman filter (KF) and the weighted robust KF



(c) Predicted data for the thresholded KF, alternative KF and weighted robust KF. All perform similarly



(d) Residual prediction error for the weighted robust Kalman filter

Fig. 2. One-dimensional data showing a cosine function with noise & outliers (and unknown system dynamics) for 500 samples at 1 sample/time step

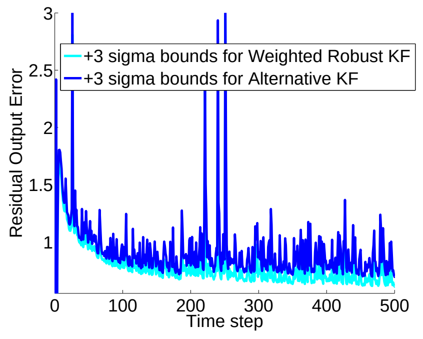


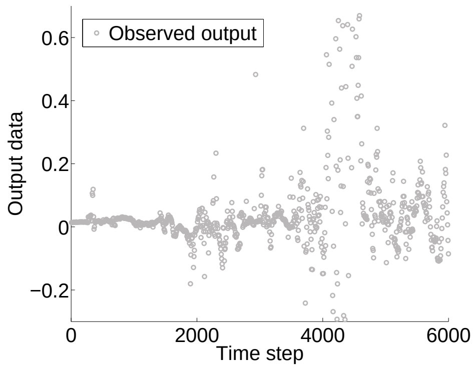
Fig. 3. $+3\sigma$ bounds for the weighted robust Kalman filter and alternative Kalman filter

3.2 LittleDog Robot

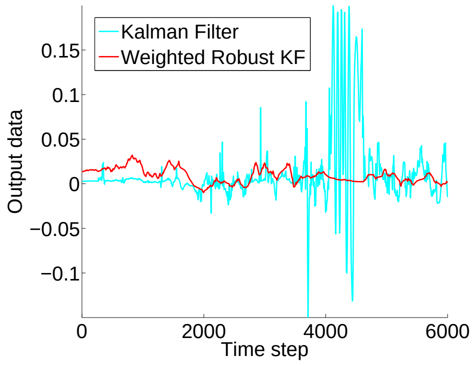
Fig. 4. LittleDog

We evaluated all filters on a 12 degree-of-freedom robotic dog, LittleDog, shown in Fig. 4. The robot dog has two sources that measure its orientation: a motion capture (MOCAP) system and an on-board inertia measurement unit (IMU). Both provide a quaternion q of the robot's orientation: q_{MOCAP} from the MOCAP and q_{IMU} from the IMU. q_{IMU} drifts over time, since the IMU cannot provide stable orientation estimation but its signal is clean. The drift that occurs in the IMU is quite common in systems where sensors collect data that need to be integrated. For example, given angular acceleration from a sensor, we may want to know what angular velocity is, and we can calculate this by integrating the angular acceleration. Unfortunately, sensor data may contain bias in the angular accleration and this bias will translate to an error in the angular velocity that will be propagated and amplified at each step an integration is done. The resulting angular velocity will have a drifting bias. In contrast, q_{MOCAP} has outliers and noise, but no drift. We would like to estimate the offset between q_{MOCAP} and q_{IMU}, and this offset is a *noisy slowly drifting signal containing outliers.* There are various approaches to estimating this slowly drifting signal, depending on the quality of estimate desired. We can estimate it with a straight line, as done in [26]. However, if we want to estimate this slowly drifting signal more accurately, we can use the proposed outlier-robust Kalman filter to track it. For optimal performance, we, once again, manually tuned **C**, **A**, **R** and **Q** for the standard Kalman filter. The system dynamics of the thresholded Kalman filter were learnt, and its threshold parameter was manually tuned for best performance on this data set.

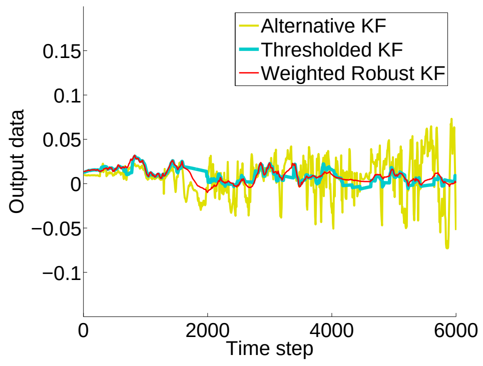
Fig. 5(a) shows the offset data between q_{MOCAP} and q_{IMU} for one of the four quaternion coefficients, collected over 6000 data samples, at 1 sample/time step. As expected, the standard Kalman filter fails to detect and ignore the outliers occurring between the 4000th and 5000th sample, as seen in Fig. 5(b). When comparing our weighted robust Kalman filter with the other remaining two filters, Fig. 5(c) shows that the thresholded Kalman filter does not react as violently as the standard Kalman filter to outliers and, in fact, appears to



(a) Observed data from LittleDog robot: a slowly drifting noisy signal with outliers



(b) Predicted data for the Kalman filter (KF) and weighted robust KF. Note the change of scale in axis from Fig. 5(a)



(c) Predicted data for the thresholded KF (which is near-optimal for this data set), alternative KF and weighted robust KF

Fig. 5. Observed vs. predicted data from LittleDog robot shown for all Kalman filters, over 6000 samples

perform similarly to the weighted robust Kalman filter. This is to be expected, given we hand-tuned the threshold parameter for optimal performance (i.e. the thresholded Kalman filter is *near-optimal* in this experiment). Notice that the weighted robust filter does not track noise in the data as closely as the alternative filter. This is a direct result of higher Kalman gains and a consequence of $\boldsymbol{\Sigma}_k$'s *explicit* dependency on $\boldsymbol{\Sigma}_{k-1}$ in the alternative filter.

In this experiment, the advantages offered by the weighted Kalman filter are clear. It outperforms the traditional Kalman filter and alternative Kalman filter, while achieving a level of performance on par with a thresholded Kalman filter (where the threshold value is manually tuned for optimal performance).

4 Conclusions

We derived a novel Kalman filter that is robust to outliers in the observations by introducing weights for each data sample. This Kalman filter learns the weights and the system dynamics, without needing manual parameter tuning by the user, heuristics or sampling. The filter performed as well as a hand-tuned Kalman filter (that required prior knowledge of the data) on real robotic data. It provides an easy-to-use competitive alternative for robust tracking of sensor data and offers a simple outlier detection mechanism that can potentially be applied to more complex, nonlinear filters.

Acknowledgments

This research was supported in part by National Science Foundation grants ECS-0325383, IIS-0312802, IIS-0082995, ECS-0326095, ANI-0224419, a NASA grant AC#98−516, an AFOSR grant on Intelligent Control, the ERATO Kawato Dynamic Brain Project funded by the Japanese Science and Technology Agency, and the ATR Computational Neuroscience Laboratories.

References

1. Moore, D.S., McCabe, G.P.: Introduction to the Practice of Statistics. W.H. Freeman & Company (March 1999)
2. Jazwinski, A.H.: Stochastic Processes and Filtering Theory. Academic Press (1970)
3. Bar-Shalom, Y., Li, X.R., Kirubarajan, T.: Estimation with Applications to Tracking and Navigation. Wiley (2001)
4. Kalman, R.E.: A new approach to linear filtering and prediction problems. In Transactions of the ASME - Journal of Basic Engineering **183** (1960) 35–45
5. Morris, J.M.: The Kalman filter: A robust estimator for some classes of linear quadratic problems. IEEE Transactions on Information Theory **22** (1976) 526–534
6. Tukey, J.W.: A survey of sampling from contaminated distributions. In Olkin, I., ed.: Contributions to Probability and Statistics. Stanford University Press (1960) 448–485

7. Huber, P.J.: Robust estimation of a location parameter. Annals of Mathematical Statistics **35** (1964) 73–101
8. Sorensen, H.W., Alspach, D.L.: Recursive Bayesian estimation using Gaussian sums. Automatica **7** (1971) 467–479
9. West, M.: Robust sequential approximate Bayesian estimation. Journal of the Royal Statistical Society, Series B **43** (1981) 157–166
10. West, M.: Aspects of Recursive Bayesian Estimation. PhD thesis, Dept. of Mathematics, University of Nottingham (1982)
11. Smith, A.F.M., West, M.: Monitoring renal transplants: an application of the multiprocess Kalman filter. Biometrics **39** (1983) 867–878
12. Meinhold, R.J., Singpurwalla, N.D.: Robustification of Kalman filter models. Journal of the American Statistical Association (1989) 479–486
13. Masreliez, C.: Approximate non-Gaussian filtering with linear state and observation relations. IEEE Transactions on Automatic Control **20** (1975) 107–110
14. Masreliez, C., Martin, R.: Robust Bayesian estimation for the linear model and robustifying the Kalman filter. IEEE Transactions on Automatic Control **22** (1977) 361–371
15. Schick, I.C., Mitter, S.K.: Robust recursive estimation in the presence of heavy-tailed observation noise. Annals of Statistics **22**(2) (1994) 1045–1080
16. Kitagawa, G.: Non-Gaussian state-space modeling of nonstationary time series. Journal of the American Statistical Association **82** (1987) 1032–1063
17. Kramer, S.C., Sorenson, H.W.: Recursive Bayesian estimation using piece-wise constant approximations. Automatica **24**(6) (1988) 789–801
18. Kitagawa, G.: Monte Carlo filter and smoother for non-Gaussian nonlinear state-space models. Journal of the American Statistical Association **93** (1996) 1203–1215
19. Kitagawa, G., Gersch, W.: Smoothness priors analysis of time series. In: Lecture Notes in Statistics. Springer-Verlag (1996)
20. Huber, P.J.: Robust Statistics. Wiley (1973)
21. Durovic, Z.M., Kovacevic, B.D.: Robust estimation with unknown noise statistics. IEEE Transactions on Automatic Control **44** (1999) 1292–1296
22. Chan, S.C., Zhang, Z.G., Tse, K.W.: A new robust Kalman filter algorithm under outliers and system uncertainties. In: IEEE International Symposium on Circuits and Systems. IEEE (2005) 4317–4320
23. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of Royal Statistical Society. Series B **39**(1) (1977) 1–38
24. Myers, K.A., Tapley, B.D.: Adaptive sequential estimation with unknown noise statistics. IEEE Transactions on Automatic Control **21** (1976) 520–523
25. Gelman, A., Carlin, J., Stern, H., Rubin, D.: Bayesian Data Analysis. Chapman and Hall (2000)
26. Ting, J., D'Souza, A., Schaal, S.: Automatic outlier detection: A Bayesian approach. In: IEEE International Conference on Robotics and Automation. (2007)
27. Ghahramani, Z., Beal, M.: Graphical models and variational methods. In Saad, D., Opper, M., eds.: Advanced Mean Field Methods - Theory and Practice. MIT Press (2000)
28. Ghahramani, Z., Hinton, G.: Parameter estimation for linear dynamical systems. Technical report, University of Toronto (1996)
29. Neal, R.M., Hinton, G.E.: A view of the EM algorithm that justifies incremental, sparse, and other variants. In Jordan, M.I., ed.: Learning in Graphical Models. MIT Press (1999) 355–368

30. Maybeck, P.S.: Stochastic models, estimation, and control. Volume 141 of Mathematics in Science and Engineering. Academic Press (1979)
31. Ting, J., D'Souza, A., Schaal, S.: Bayesian regression with input noise for high dimensional data. In: Proceedings of the 23rd International Conference on Machine Learning, ACM (2006) 937–944