Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

Article

Deep Kalman Filter: Simultaneous Multi-Sensor Integration and Modelling; GNSS/IMU Case Study

Siavash Hosseinyalamdary [1]

[1] Department of Earth Observation Science (EOS), Faculty of Geo-information Science and Earth Observation (ITC), University of Twente; s.hosseinyalamdary@utwente.nl; e-mail@e-mail.com; Tel.: +31-(0)53-489-3921

Abstract: The Bayes filters, such as Kalman and particle filters, have been used in sensor fusion to integrate two sources of information and obtain the best estimate of the unknowns. Efficient integration of multiple sensors requires deep knowledge of their error sources and it is not trivial for complicated sensors, such as Inertial Measurement Unit (IMU). Therefore, IMU error modelling and efficient integration of IMU and Global Navigation Satellite System (GNSS) observations has remained a challenge.

In this paper, we develop deep Kalman filter to model and remove IMU errors and consequently, improve the accuracy of IMU positioning. In other words, we add modelling step to the prediction and update steps of Kalman filter and the IMU error model is learned during integration. Therefore, our deep Kalman filter outperforms Kalman filter and reaches higher accuracy.

Keywords: Deep Kalman Filter; Simultaneous Sensor Integration and Modelling (SSIM); GNSS/IMU Integration; Recurrent Neural Network; Deep Learning; Long-Short Term Memory (LSTM);

1. Problem Statement

Global Navigation Satellite System (GNSS) enables us to locate ourselves within a few millimeters all over the world. This system consists of Global Positioning System (GPS), Galileo, Glonass, and Beidu, and it is integrated in our daily life from car navigators to airplanes. Unfortunately, GNSS positioning requires clear sky view and therefore, it is not available in urban canyons where GNSS signals are blocked by high-rise buildings. Therefore, other alternative navigation solutions are applied to overcome this shortcoming of GNSS positioning and bridge its gaps in urban canyons.

Among alternative navigation solutions, inertial navigation and visual odometry are cost-effective and they do not require any infrastructure. Inertial Measurement Unit (IMU) is a composition of accelerometers and gyroscopes and it estimates position, velocity, and orientation of the platform from measured accelerations and angular rates.

The error characteristics of IMU sensors are complicated. They significantly differ from one technology, one manufacturer, and even one sensor to another. The cold atom IMUs are the most accurate IMUs (Battelier 2016), but they are very expensive and they are not applied for commercial applications such as mobile mapping. Other IMU technologies, such as Mechanical Gyro, Ring Laser Gyro (LRG), Fiber Optic Gyro (FOG), and Micro-Electro-Mechanical System (MEMS), suffer from common error sources, such as bias and scale factor errors, but some technologies have their own specific error sources, such as dead zone error in RLG. MEMS sensors, which are cost-effective and are frequently used in industry, suffer from various error sources and their positioning accuracy quickly deteriorates in the absence of GNSS positioning.

The IMU sensors have systematic, random, and computational error sources. The IMU manufacturers use controlled environments, such as turntable, to estimate and remove systematic

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

errors. The calibration procedure using controlled environment is costly and it cannot fully remove systematic errors. Some random errors, such as run-to-run bias, are added to the state vector and are estimated and removed from IMU observations when GNSS positioning is available. Therefore, these error sources should be modelled. The true nature of these errors is very complicated and it depends on many unobserved parameters, such as temperature.

In contrast, the Bayes estimators, such as the Kalman and Particle filters, have many limitations and they may not be able to model the IMU errors. In the Kalman filter, the error model should linear with Gaussian distribution and therefore, they are not very useful for the IMU error modeling. Particle filter can handle non-linear IMU error models with non-Gaussian distribution. However, particle filter is applicable when the IMU error model and its data distribution are known.

If the error characteristics of IMU sensors can be accurately modelled, its accuracy is significantly improved. Currently, scientists suggest different error models for IMU error sources separately. For instance, Jekeli (2001) has suggested to model the bias of accelerometer and gyroscope as random constant and Noureldin et al. (2013) has applied first-order Gauss-Markov stochastic process to model these biases.

In this paper, we introduce deep Kalman filter to simultaneously integrate GNSS and IMU sensors and model the IMU errors. In contrast to previously proposed approaches, our approach does not have any pre-defined IMU error model and it is learned from observations. Therefore, we do not need to assume any stochastic or deterministic behavior of IMU errors. In contrast to previously proposed approaches, we can accurately model the non-linear, time-variant, highly correlated IMU error sources.

1.1. Literature Review

There are a few scientific endeavors to harness the IMU errors and provide accurate alternative positioning system in the absence of GNSS positioning. We have already discussed that bias of accelerometer and gyro is modelled as random constant in Jekeli (2001) and it is modelled as first-order Gauss-Markov stochastic process in Noureldin et al. (2013). Shin and El-Sheimy (2002) use gravity and earth rotation rate to calibrate the IMU errors.

Wang et al. (2016) treat the IMU error sources as time series and apply the AutoRegressive and Moving Average (ARMA) to model the IMU error sources. In addition to the statistical estimators, shallow Multi-Layer Perceptron (MLP) network have been utilized to model IMU error sources in the presence of GNSS positioning (Chiang et al. 2008, Noureldin et al. 2011). Adaptive Neural Fuzzy Information Systems (ANFIS) has also applied to capture the IMU uncertainties (Abdel-Hamid et al. 2007, Zhang et al. 2014). Toth and his colleagues have applied neural networks and fuzzy logic (Toth et al. 2007) to model the IMU error sources. Navidi et al. (2015) use hybrid Fuzzy Inference System (FIS) and second-order extended Kalman filter to integrate GNSS and IMU sensors. Xing and his colleagues have showed that chaotic particle swarm optimization significantly reduces the gyroscope random drift (Xing et al 2017). The applied neural networks are shallow and they do not accurately calibrate IMU since they do not consider the IMU's correlation over time. In other words, IMU error sources should be studied as a time series and model the IMU error sources over time.

It has been shown that the shallow neural networks can only model simple phenomena and the complicated systems should be modelled using deep neural network (Minsky and Papert 1988). Therefore, we apply deep neural network for sensor integration. Up to our knowledge, we are the first who apply deep neural network for GNSS and IMU integration and IMU error modelling.

Mirowski and Lecun (2009) have introduced dynamic factor graphs and reformulated the Bayes filters as recurrent neural networks. In their proposed approach, the observation and system models of the Kalman filter are learned from observations. Gu et al. (2017) reformulate Kalman filter and recurrent neural network to model face landmark localization in videos. Krishnan et al. (2015) integrate Kalman filter and variation methods for learning famous machine learning dataset, MNIST.

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

2. Kalman filter

The unknown vector, which is estimated in the Kalman filter, is called state vector and it is represented by $\mathbf{x}_t \in \mathbb{R}^n$, where t indicates to the state vector at time t. It also depends on the observation vectors, $\mathbf{z}_{1:t}$, where $\mathbf{z}_t \in \mathbb{R}^m$, and the initial state of the system $\mathbf{x}_0$. The probability of state vector at the current time is $\Pr(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{x}_0)$. Therefore, the current state vector is estimated using Maximum Likelihood (ML), such that:

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \Pr(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{x}_0) \tag{1}$$

The probability of current state vector depends on the previous state vectors, such that

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \frac{\Pr(\mathbf{z}_t|\mathbf{x}_t) \Pr(\mathbf{x}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_0)}{\Pr(\mathbf{z}_{1:t})} \tag{2}$$

The denominator in Equation (2) is a normalization constant and it does not contribute in maximization of likelihood in Equation (2), such that:

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \Pr(\mathbf{z}_t|\mathbf{x}_t) \Pr(\mathbf{x}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_0) \tag{3}$$

The state vector at current time directly depends on the previous state vectors. Therefore, the marginalization of previous state vectors is applied, such that:

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \Pr(\mathbf{z}_t|\mathbf{x}_t) \int \Pr(\mathbf{x}_t|\mathbf{x}_{1:t-1}) \Pr(\mathbf{x}_{1:t-1}|\mathbf{z}_{1:t-1}, \mathbf{x}_0) d\mathbf{x}_{1:t-1} \tag{4}$$

Based on Markovian assumption, the state vector at the current time only depends on the state vector at the previous time, such that:

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \Pr(\mathbf{z}_t|\mathbf{x}_t) \int \Pr(\mathbf{x}_t|\mathbf{x}_{t-1}) \Pr(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}, \mathbf{x}_0) d\mathbf{x}_{t-1} \tag{5}$$

where $\Pr(\mathbf{z}_t|\mathbf{x}_t)$ is the posterior probability estimation of the current state vector. The state vector best estimate in the previous time is $\hat{\mathbf{x}}_{t-1} = \underset{\mathbf{x}_{t-1}}{\operatorname{argmax}} \Pr(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}, \mathbf{x}_0)$. Therefore, equation (5) is reformulated as:

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}} \Pr(\mathbf{z}_t|\mathbf{x}_t) \int \Pr(\mathbf{x}_t|\mathbf{x}_{t-1}) \hat{\mathbf{x}}_{t-1} d\mathbf{x}_{t-1} \tag{6}$$

where $\Pr(\mathbf{x}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_0) = \int \Pr(\mathbf{x}_t|\mathbf{x}_{t-1}) \hat{\mathbf{x}}_{t-1} d\mathbf{x}_{t-1}$ is the prior probability estimation. It predicts the current state vector based on the system model and the posterior estimation of state vector in the previous time.

In the Kalman filter, the state vector is related to the state vector at the previous times, $t-1$ using system model, f, such that:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \epsilon_t \tag{7}$$

where, ϵ_t, is noise of system model. The state vector relates to the observation vector, $\mathbf{z}_t \in R^m$, with the observation model, g, such that:

$$\mathbf{z}_t = g(\mathbf{x}_t) + \omega_t \tag{8}$$

where, ω_t, is noise of observation model. The state and observation models are assumed to be linear in the Kalman filter. Therefore, these functions can be replaced by $\mathtt{F}$ and $\mathtt{G}$ matrices, respectively. The system model is rewritten, such that:

$$\mathbf{x}_t = \mathtt{F}\mathbf{x}_{t-1} + \epsilon_t \tag{9}$$

and similarly, the observation model is rewritten, such that:

$$\mathbf{z}_t = \mathtt{G}\mathbf{x}_t + \omega_t \tag{10}$$

Noise of system and observation models have Normal distribution in the Kalman filter, such that:

$$\epsilon_t \sim \mathcal{N}(0, \mathtt{Q}_t) \tag{11}$$

$$\omega_t \sim \mathcal{N}(0, \mathtt{R}_t) \tag{12}$$

where $\mathtt{Q}_t$ and $\mathtt{R}_t$ are the covariance matrices of system and observation models.

The Kalman filter is designed in the two-stage optimization. In the first stage, the current state vector is predicted based on the state vector in the previous time, such that:

$$\mathbf{x}_t^- = \mathtt{F}\mathbf{x}_{t-1}^+ \tag{13}$$

The predicted values are represented by superscript '-' and the updated values are represented by superscript '+'. The error propagation is applied to estimate the covariance matrix of current state vector based on the covariance matrix of state vector in the previous time, such that:

$$\mathtt{P}_t^- = \mathtt{F}\mathtt{P}_{t-1}^+\mathtt{F}^\top + \mathtt{Q}_t \tag{14}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 March 2018 doi:10.20944/preprints201803.0121.v1

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

where P_t^- is the predicted covariance matrix of state vector. In the update stage of Kalman filter, the current state vector is updated by using observation vector, such that:

$$\mathbf{x}_t^+ = \mathbf{x}_t^- + K_t(\mathbf{z}_t - G_t\mathbf{x}_t^-) \tag{15}$$

and the covariance matrix of updated current state vector is calculated, such that:

$$P_t^+ = (I - K_tG_t)^\top P_t^-(I - K_tG_t) + K_tR_tK_t \tag{16}$$

where K_t is the Kalman gain matrix and it is calculated, such that

$$K_t = P_t^-G_t^\top(G_tP_t^-G_t^\top + R_t)^{-1} \tag{17}$$

For derivation of Kalman filter equations, reader is referred to Jeleki (2001). Figure 1 shows the Kalman filter.

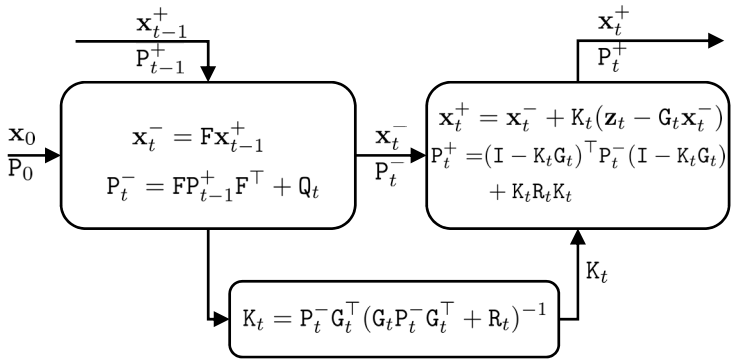


Figure 1: The Kalman filter procedure; It consists of prediction (left-up box) and update steps (right-up box).

2.1.1. Shortcoming of Kalman filter

There are a number of shortcomings in Kalman filter: F and G are linear models with Gaussian noise. Therefore, they cannot model non-linear functions or linear functions with non-Gaussian noise. In addition, these functions are time invariant and they do not change over time. A number of variations in Kalman filter, such as extended and unscented Kalman filters, can handle non-linear observation and system models. In addition, particle filter works on the observation and system models with other distributions than Gaussian noise.

Nonetheless, the observation and system models of Bayes filters should be known. In other words, scientists should find a model to relate state vectors to observations and the dynamic of the system. For instance, we discussed that the bias of accelerometer and gyro are modelled in different ways. In addition, noise probability distribution of the observation and system model should be known. Unfortunately, the observation and system models cannot be determined beforehand in many applications.

Another shortcoming of the Bayes filters is their Markovian assumption. In Bayes filter, it is assumed that the current state vector only depends on the previous state vector and it is independent from older state vectors. Although, this property significantly simplifies the system model and it makes these filters very efficient, it makes the Bayes filter insensitive to system behavior with longer correlation time. In other words, complicated error models with long correlation time cannot be modelled in Bayes filters.

Here, we provide our discussion with an example on our case study, IMU error modelling. The IMU models have different error sources depending the technology, which is applied in their accelerometers and gyroscopes. One IMU manufacturer can calibrate the IMU error sources different from another manufacturer. Moreover, the error sources of MEMS sensors can significantly differ from one MEMS sensor to another even in one family. The pre-defined IMU error models cannot handle high variations of error sources in MEMS sensors. In addition, high correlation between different MEMS error sources makes the IMU error modelling very complicated. Often, it is not possible to discriminate the error sources in MEMS sensors.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 March 2018 doi:10.20944/preprints201803.0121.v1

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

3. Methodology

In this paper, we add system modelling to the Kalman filter and we call it deep Kalman filter. In other words, deep Kalman filter is able to estimate the system model and it is useful in many applications, such as GNSS/IMU integration, where the system model is complicated. In order to estimate the system model of Kalman filter, we add latent variables to the Kalman filter. Latent variables are not observed and they are invisible in the state vector, but the state vector depends on the latent variables. As an example, IMU errors depend on the temperature, but it is not observed and cannot be estimated. These variables are represented with the latent vector, $\mathbf{h}_t$, where the subscription t stands for the latent vector in the current time.

We assume the current state vector, $\mathbf{x}_t$, depends on the current and previous latent vectors, $\mathbf{h}_{t:t-T}$. Therefore, the current state vector indirectly depends on previous state vectors, $\mathbf{x}_{t-1:t-T}$, and Markovian assumption does not hold anymore. We design our modelling step in the way that the current state vector depends on the current latent vector and current latent vector depends on previous state vectors and previous latent vectors. This is one of many different architectures of possible networks, but it significantly simplifies our network.

Let's assume there is a function ϕ that relates the current latent vector to the previous latent vectors and previous state vectors, such that:

$$\mathbf{h}_t = \phi(\mathbf{x}^+_{t-1:t-T}, \mathbf{h}_{t-1:t-T}) \tag{18}$$

and the current state vector is directly related to the current latent vector by a function, λ, such that:

$$\mathbf{x}^{+-}_t = \lambda(\mathbf{h}_t) + \mu_t \tag{19}$$

we name the posterior estimation based on our model as $\mathbf{x}^{+-}_t$. In other words, $\mathbf{x}^{+-}_t$ is the predicted posterior estimation of state vector. Functions ϕ and λ are a combination of linear and non-linear functions and therefore, they can create non-linear models. The linear function has several coefficient parameters, represented by coefficient matrices, $\mathbb{W}$, but the non-linear function, σ, has no parameter. Therefore, our network is designed, such that:

$$\mathbf{h}_t = \sigma(\mathbb{W}_{xh}\mathbf{x}^+_{t-1:t-T} + \mathbb{W}_{hh}\mathbf{h}_{t-1:t-T}) \tag{20}$$

$$\mathbf{x}^{+-}_t = \sigma(\mathbb{W}_{xx}\mathbf{h}_t) + \mu_t \tag{21}$$

Let's name the parameters of latent vector, $\mathbb{W}_h = [\mathbb{W}_{xh}, \mathbb{W}_{hh}]$. In order to model our system, it suffices to estimate $\mathbb{W}_h$ and $\mathbb{W}_{xx}$. Figure 2 shows the probabilistic graphical model of Kalman filter and deep Kalman filter. The upper part of deep Kalman filter is prediction and update steps and it is similar to Kalman filter. However, we added modelling step to Kalman filter and it is the lower part of deep Kalman filter in Figure 2.

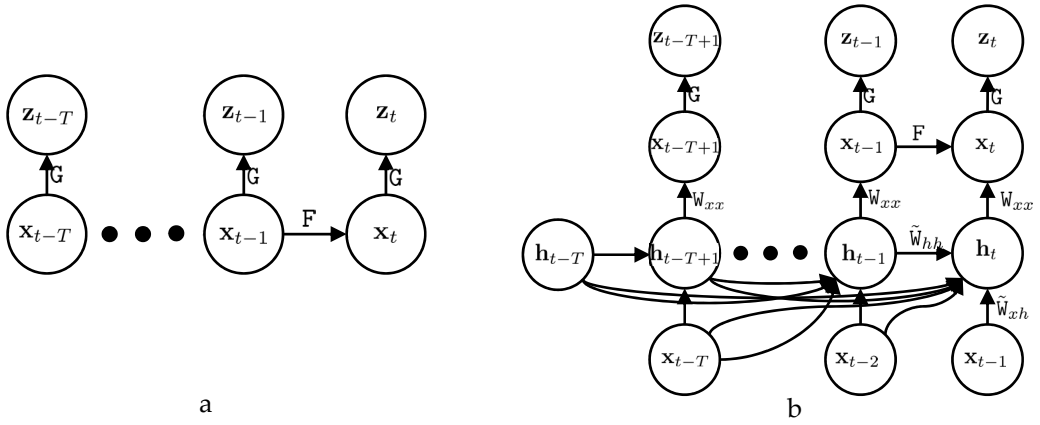


Figure 2: The probabilistic graphical model of Kalman filter (a) and deep Kalman filter (b).

When the posterior estimation of state vector is available, $\mathbf{x}^+_t$, it can be replaced with $\mathbf{x}^{+-}_t$ in equation (21) and the latent vector and the coefficient matrix, $\mathbb{W}_{xx}$, are estimated. In order to estimate the latent vector, we use maximum likelihood estimation. In other words, we maximize the likelihood of latent vector, such that:

$$\hat{\mathbf{h}}_t = \underset{\mathbf{h}_t}{\operatorname{argmax}} \Pr(\mathbf{x}^+_t | \mathbf{h}_t, \mathbf{W}_{xx}) \tag{22}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 March 2018 doi:10.20944/preprints201803.0121.v1

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

Instead of latent vector estimation, it suffices to estimate coefficient matrix, $\mathtt{W}_h$. By replacing latent vector from equation (20) into equation (22), it is reformulated as:

$$\hat{\mathtt{W}}_h = \underset{\mathtt{W}_h}{\operatorname{argmax}} \Pr\left(\mathbf{x}_t^+ | \mathtt{W}_h, \mathtt{W}_{xx}, \mathbf{x}_{t-1:t-T}^+, \mathbf{h}_{t-1:t-T}\right) \tag{23}$$

The latent vector on the left side depends on the coefficient matrix, $\mathtt{W}_{xx}$, in the right side, which is also unknown. Therefore, we utilize expectation-maximization iterative method to estimate all coefficient matrices, and consequently model our system.

In the expectation maximization, we first assume there is an initial guess for our coefficient matrix $\mathtt{W}_{xx}^{(0)}$. Based on the initial guess of coefficient matrix, the latent vector, $\hat{\mathtt{W}}_h^{(1)}$, is estimated, such that:

$$\hat{\mathtt{W}}_h^{(1)} = \underset{\mathtt{W}_h}{\operatorname{argmax}} \Pr\left(\mathbf{x}_t^+ | \mathtt{W}_h, \mathtt{W}_{xx}^{(0)}, \mathbf{x}_{t-1:t-T}^+, \mathbf{h}_{t-1:t-T}\right) \tag{24}$$

where superscript numbers with parenthesis indicate to iterations.

The new coefficient matrix of $\mathtt{W}_{xx}^{(1)}$ is estimated by maximizing the probability of coefficient matrix based on the latent vectors, such that:

$$\hat{\mathtt{W}}_{xx}^{(1)} = \underset{\mathtt{W}_{xx}}{\operatorname{argmax}} \Pr\left(\mathbf{x}_t^+ | \mathtt{W}_h^{(1)}, \mathtt{W}_{xx}, \mathbf{x}_{t-1:t-T}^+, \mathbf{h}_{t-1:t-T}\right) \tag{25}$$

The iteration continues until it converges to local maximum and $[\mathtt{W}_{xx}^{(m)}, \mathtt{W}_h^{(m)}]$ and $[\mathtt{W}_{xx}^{(m+1)}, \mathtt{W}_h^{(m+1)}]$ do not significantly change. When the coefficient matrices are estimated, the system model is estimated. After the system model is estimated, the state vector can be predicted using the modelled system in equations (20) and (21). Figure 3 shows the scheme of deep Kalman filter.

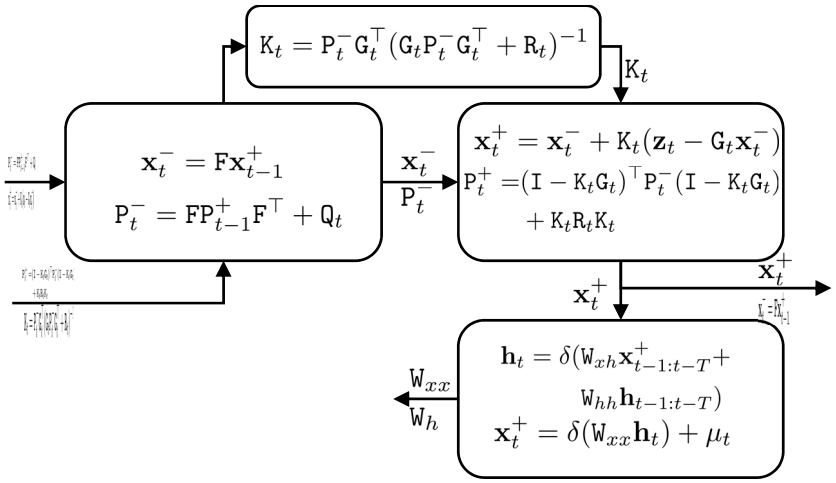


Figure 3: The deep Kalman filter procedure. The IMU modelling step (right-bottom) has been added to Kalman filter.

3.1. Recurrent Neural Network

The expectation maximization will lead to global maximum in the convex functions. However, it is most likely it converges to local maximum, since modelling is not convex for complicated models. One of the challenges is finding an approach to maximize the equations (24) and (25). This maximization is equivalent to minimization of model prediction and the current state vector. Therefore, we can utilize gradient descent to minimize the error in prediction using our model. In order to minimize the prediction error, we define an energy function, E, and minimize it, such that:

$$E(\mathtt{W}_h, \mathtt{W}_{xx}) = \frac{1}{2}(\mathbf{x}_t^{+-} - \mathbf{x}_t^+)^2 \tag{26}$$

by replacing $\mathbf{x}_t^{+-}$ from equation (21), equation (26) is reformulated, such that:

$$E(\mathtt{W}_h, \mathtt{W}_{xx}) = \frac{1}{2}(\sigma(\mathtt{W}_{xx}\mathbf{h}_t) - \mathbf{x}_t^+)^2 \tag{27}$$

Therefore, the gradient of coefficient matrix, $\mathtt{W}_{xx}$, is calculated, such that:

$$\frac{\partial E(\mathtt{W}_h, \mathtt{W}_{xx})}{\partial \mathtt{W}_{xx}} = \frac{\partial \sigma(\mathtt{W}_{xx}\mathbf{h}_t)}{\partial \mathtt{W}_{xx}}(\sigma(\mathtt{W}_{xx}\mathbf{h}_t) - \mathbf{x}_t^+) \tag{28}$$

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

The gradient of latent vector coefficient matrix, $\mathbb{W}_h$, is also estimated, such that:

$$\frac{\partial E(\mathbb{W}_h, \mathbb{W}_{xx})}{\partial \mathbb{W}_h} = \frac{\partial \sigma(\mathbb{W}_{xh}\mathbf{x}^+_{t-1:t-T} + \mathbb{W}_{hh}\mathbf{h}_{t-1:t-T})}{\partial \mathbb{W}_h} \frac{\partial \sigma(\mathbb{W}_{xx}\mathbf{h}_t)}{\partial \mathbb{W}_{xx}} (\sigma(\mathbb{W}_{xx}\mathbf{h}_t) - \mathbf{x}^+_t) \quad (29)$$

Therefore, the coefficient matrices of modelled system, $\mathbb{W}_{xx}$ and $\mathbb{W}_h$, are estimated in the iterative gradient descent approach, such that:

$$\mathbb{W}^{(m+1)}_{xx} = -\frac{\partial E(\mathbb{W}_h, \mathbb{W}_{xx})}{\partial \mathbb{W}_{xx}} \mu + \mathbb{W}^{(m)}_{xx} \quad (30)$$

$$\mathbb{W}^{(m+1)}_{h} = -\frac{\partial E(\mathbb{W}_h, \mathbb{W}_{xx})}{\partial \mathbb{W}_{h}} \mu + \mathbb{W}^{(m)}_{h} \quad (31)$$

where μ is the learning rate. When the coefficient matrices are determined and system model is learned, $\mathbf{x}^{+-}_t$ can be estimated based on previous state vectors, $\mathbf{x}^+_{t-1:t-T}$, in equations (20) and (21).

3.2. Long Short Term Memory

Recurrent neural networks have a drawback, known as exploding-vanishing gradient problem (Goodfellow et al. 2016). When T is large and we model the system for a long time span, the gradients are multiplied in several layers. If the gradients are large, their multiplication becomes humongous and their gradient explodes. In the contrary, if the gradients are small, their multiplication becomes insignificant, known as vanishing gradient. In order to prevent such an effect in recurrent neural networks, Long Short Term Memory (LSTM) has introduced gated memories (Hochreiter and Schmidhuber 1997). In LSTM, the network consists of cells and each cell can memorize the previous state vectors, using input gate, remember them, using output gate, and forget them, using forget gate. Let's represent input gates as $\mathbf{i}_t$, output gates as $\mathbf{o}_t$, , and forget gates as $\mathbf{f}_t$. They are represented by a combination of linear and non-linear functions, such that:

$$\mathbf{f}_t = \sigma(\mathbb{W}_f\mathbf{x}_t + \mathbb{U}_f\mathbf{h}_{t-1}) \quad (32)$$

$$\mathbf{i}_t = \sigma(\mathbb{W}_i\mathbf{x}_t + \mathbb{U}_i\mathbf{h}_{t-1}) \quad (33)$$

$$\mathbf{o}_t = \sigma(\mathbb{W}_o\mathbf{x}_t + \mathbb{U}_o\mathbf{h}_{t-1}) \quad (34)$$

where σ is the non-linear function and the linear functions are represented by coefficient matrices, $\mathbb{W}_f$, $\mathbb{W}_i$, and $\mathbb{W}_o$. The cell state, $\mathbf{c}_t$, and hidden layer, $\mathbf{h}_t$, are estimated, such that:

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \sigma(\mathbb{W}_c\mathbf{x}_t + \mathbb{U}_c\mathbf{h}_{t-1}) \quad (35)$$

$$\mathbf{h}_t = \mathbf{o}_t\sigma(\mathbf{c}_t) \quad (36)$$

where $\circ$ is Hadamard product. For long term correlation, input gate can keep information from previous state vectors and the gradients of previous state vectors are accessible. Therefore, the gradients do not explode or vanish in the back-propagation process. The forget gate controls the complexity of the model and it removes the uncorrelated previous state vectors.

4. Implementation

In this chapter, we explain the details of our implementation. It has been shown that the Kalman filter works better on IMU errors instead of IMU output (Jekeli 2001, Noureldin et al. 2011). Therefore, we utilize the IMU mechanization to estimate position, velocity, and orientation. However, IMU mechanization is not perfect and the error of position, velocity, and orientation remains in the system. We define the state vector of Kalman filter consists of positioning error, velocity error, orientation error, and bias of accelerometers and gyroscopes. The state vector is estimated in an extended Kalman filter when GNSS observations are available and Extended Kalman filter predicts the state vector in the absence of GNSS signals. The system model utilized in the Kalman filter is similar to Noureldin et al. (2011).

When GNSS signals are available the posterior estimation of state vectors, $\mathbf{x}^+_t$, is calculated. Since we use Real-Time Kinematic (RTK) technique for GNSS observation, our posterior estimation will be very accurate and it can be used as ground truth for our IMU error modelling. Let's call the predicted posterior estimation of IMU errors using our modelling approach as $\mathbf{x}^{+-}_t$. We try to predict $\mathbf{x}^{+-}_t$ as close as possible to $\mathbf{x}^+_t$ and therefore, we get the best estimate of modelled IMU errors. In other words, we find a model to calculate $\mathbf{x}^{+-}_t$ as an approximation of $\mathbf{x}^+_t$ in the presence of GNSS solutions and we replace $\mathbf{x}^-_t$ with $\mathbf{x}^{+-}_t$ in the absence of GNSS solution. If the system model is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 March 2018 doi:10.20944/preprints201803.0121.v1

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

accurately estimated, the predicted state vector using our model, $\mathbf{x}_t^{+-}$, has higher accuracy than the predicted state vector $\mathbf{x}_t^{-}$.

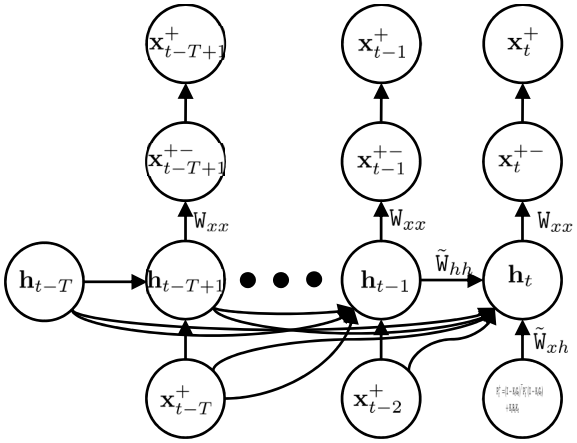


Figure 4: The IMU error modelling is reformulated as a time series prediction.

In order to implement an efficient predictive recurrent neural network, we have designed a network with only one hidden layer while every hidden vector, $\boldsymbol{h}_t$, contains 3000 variables, known as nodes in neural network terminology. We are benefited from the mini-batch learning where the sequences are added together and processed once. The batch size of our networks is 800. It speeds up the learning process and improves the learning convergence. Another important factor of training in recurrent neural network is the learning rate. We use 0.05 as initial learning rate with 5e-3 learning rate decay. We have tried several sequence lengths and studied the impact of the sequence length on accurate error modeling.

When GNSS observations are not available, the posterior estimation of state vector cannot be calculated. Therefore, $\mathbf{x}_t^{+}$ is not estimable and we can only predict the state vector. There are two ways to predict the current state vector: using system model of Kalman filter or applying the modelled system model of IMU in the deep Kalman filter. We anticipate modelled system in the deep Kalman filter has better performance over system model of Kalman filter. In other words, the modelled IMU errors should be more accurate than the predicted IMU errors of Kalman filter.

5. Experiment

In order to evaluate our proposed approach, KITTI benchmark (Geiger et al. 2013) has been utilized. This benchmark contains multi-sensor datasets and labeled information. Among various sensors, GPS and IMU information have been used in this paper. We use the longest KITTI dataset, dataset No. 34, collected on October 3, 2011. This dataset contains 7:46 minutes of data collection, mostly driven in residential area with clear sky view. The trajectory of platform exceeds 1.7 kilometers, as shown in Figure 5. The OXTS RT 3003 navigation system has been applied to collect the GPS and IMU information. The GPS and IMU have already been synchronized and have been collected in 10 Hz. Manufacturers have performed calibration procedure and the calibration parameters, such as level-arm, are internally applied to transfer GPS information into the IMU local frame. The dual-frequency RTK has been used to estimate accurate position of IMU with the accuracy better than 10 centimeters.

We use the GPS observations as ground truth to train our network and model the IMU errors. When the weights of our network are learned and IMU errors are modelled, we estimate the IMU errors in two ways and compare them: First, we do not use GPS observations and we use different approaches to predict IMU errors and correct IMU Positioning. Second, we use GPS observations and calculate the IMU errors. The calculated IMU errors using GPS observations are utilized to correct IMU positioning and we use it as baseline to evaluate the corrected IMU positioing without using GPS observations.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 March 2018 doi:10.20944/preprints201803.0121.v1

Peer-reviewed version available at Sensors 2018, 18, 1316; doi:10.3390/s18051316

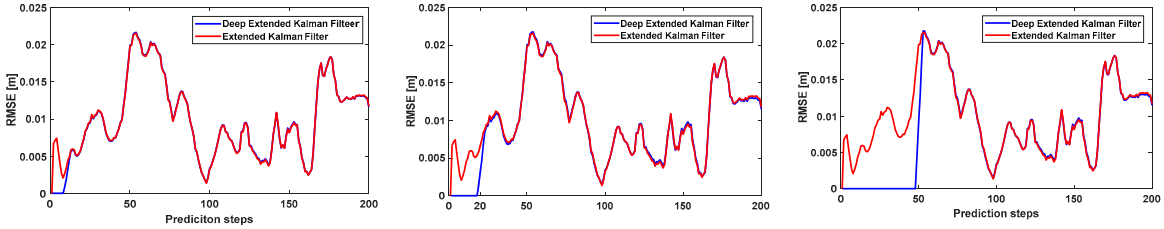
Figure 5: The trajectory of KITTI dataset, #34. It is the longest KITTI dataset where the vehicle travels more than 1.7 kilometers.

6. Results

In order to evaluate the results, we divide the trajectory into two parts, one for training and calibration and one for testing and evaluation. In the training phase, the parameters of system model are estimated in the extended Kalman filter, as described in Noureldin et al. (2011). In the deep extended Kalman filter, IMU errors are modelled in addition to the prediction and update stages. In other words, we train our network to predict the posterior estimation of IMU errors using recurrent neural network and consequently, model the IMU errors. When the network is trained, we should be able to predict the posterior estimation of IMU errors and correct the IMU positioning.

In the testing part, we estimate the IMU errors using GNSS observations and correct the IMU positioning. We utilize the corrected IMU positioning as ground truth. The extended Kalman filter and deep extended Kalman filter are applied to predict the IMU errors without using GNSS observations and correct the IMU positioning. The corrected IMU positioning using these two approaches are compared with the ground truth and the IMU positioning error is calculated. The Root Mean Square Error (RMSE) of IMU positioning is applied to evaluate the results of extended Kalman filter and deep extended Kalman filter.

In the first experiment, we used simple RNN to predict the IMU errors and remove them from the IMU positioning. In this experiment, we study the performance of extended Kalman filter and deep extended Kalman filter using different sequence lengths. In other words, the sequences of IMU errors, utilized to predict the IMU errors in the RNN have different lengths. We have plotted the RNN in Figure 6 for the sequence length of 10, 20, and 50.



a) simple RNN with sequence length of 10

b) simple RNN with sequence length of 20

c) simple RNN with sequence length of 50

Figure 6: The RMSE of deep extended Kalman filter and extended Kalman filter (Hosseinyalamdary and Balazadegan 2017); We used different sequence lengths of simple RNN for IMU modelling of deep extended Kalman filter.

Figure 6 shows that the performance of RNN depends on the sequence length of previous state vectors. If the sequence length is large, the network uses more IMU errors in the previous time. In other words, the network applies more heuristics to predict the IMU errors and remove them from IMU positioning. Therefore, the RMSE of deep Extended Kalman filter is less than the RMSE of extended Kalman filter in the early time.

Despite of the effectiveness of RNN in the prediction of IMU errors in short period, it cannot predict IMU errors in the long period of time. In other words, the RMSE of deep extended Kalman filter is lower than the RMSE of extended Kalman filter in earlier time, but two approaches have the same RMSE in longer period of time. This effect is due to vanishing-exploding gradient problem of RNN. This well-known problem of RNN, has overcome by the development of Long Short Term Memory (LSTM). LSTM uses memory gates to remember to previous IMU errors and prevents repetitive gradient multiplications.

Figure 7 shows the IMU positioning using LSTM network with the sequence length of 10. In othr words, instead of using RNN to predict IMU errors, we use LSTM to predict IMU errors and remove them from IMU positioning. The sequence length of this network is similar to Figure 6a. However, the effectiveness of this network does not disappear in the longer period of time. In Figure 7, deep extended Kalman filter outperforms the extended Kalman filter for most of the time. There are instances that extended Kalman filter has better accuracy than the deep extended Kalman filter, but they are limited to few small instances. The total RMSE of deep extended Kalman filter is 0.0100 meters and the total RMSE of extended Kalman filter is 0.0114 meters for 20 seconds. In summary, deep extended Kalman filter has better accuracy over extended Kalman filter.

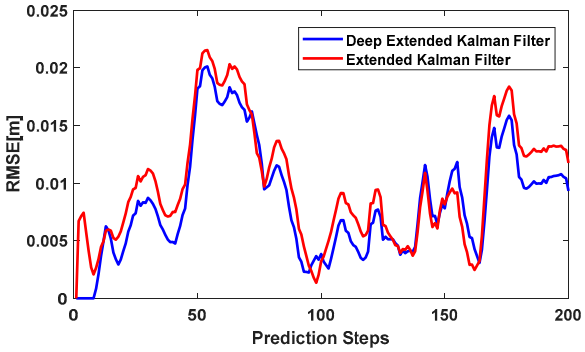


Figure 7: The RMSE of deep extended Kalman filter and extended Kalman filter; deep extended Kalman filter IMU modelling is based on LSTM with sequence length of 10.

7. Conclusions

In this paper, we have introduced deep Kalman filter to estimate the system model of Kalman filter. In other words, in addition to the prediction and update steps of the Kalman filter, we have added modelling step to the deep Kalman filter. We have applied the deep Kalman filter to model the IMU errors and correct the IMU positioning. In the deep Kalman fitler, the model of IMU errors is learned using Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) when GNSS observations are available. The IMU errors can be predicted using learned model in the absence of GNSS observations.

We have experimented the Kalman filter and deep Kalman filter using KITTI dataset and the results show that our approach outperforms the Kalman filter. Therefore, we reach better accuracy in the deep Kalman filter. In addition, deep Kalman filter using RNN can predict the IMU errors for short period of time, but its effectiveness disappears in longer period of time due to its vanishing-exploding gradient problem. Deep Kalman filter based on Long Short Term Memory (LSTM) have better accuracy over Kalman filter and it can predict IMU errors in longer period of time.

References

1. Jekeli C. (2001), Inertial Navigation Systems with Geodetic Applications, Walter de Gruyter.
2. Noureldin A., Karamat T.B., Georgy J. (2013), Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration, Springer-Verlag Berlin Heidelberg.
3. Shin E.-H. and El-Sheimy N. (2002), A new calibration method for strap down inertial navigation systems. Zeitschrift f¨ur Vermessungswesen, 127 (1), 1–10.
4. Wang, S., Deng, Z., & Yin, G. (2016), An Accurate GPS-IMU/DR Data Fusion Method for Driverless Car Based on a Set of Predictive Models and Grid Constraints. Sensors (Basel, Switzerland), 16(3), 280.

5. Chiang K.W., Noureldin A., El-Sheimy N. (2008), Constructive Neural-Networks-Based MEMS/GPS Integration Scheme, in IEEE Transactions on Aerospace and Electronic Systems, vol. 44, no. 2, pp. 582-594.
6. Noureldin A., El-Shafie A., Bayoumi M. (2011), GPS/INS Integration Utilizing Dynamic Neural Networks for Vehicular Navigation, Information Fusion 12 (1), pp. 48-57.
7. Abdel-Hamid W., Noureldin A., El-Sheimy N., (2007), Adaptive Fuzzy Prediction of Low-Cost Inertial-Based Positioning Errors, in IEEE Transactions on Fuzzy Systems, vol. 15, no. 3, pp. 519-529.
8. Zhang L., Liu J., Lai J., Xiong Z., (2014) Performance Analysis of Adaptive Neuro Fuzzy Inference System Control for MEMS Navigation System, Mathematical Problems in Engineering, Vol 2014.
9. C. Toth, D. A. Grejner-Brzezinska, and S. Moafipoor, "Pedestrian Tracking and Navigation Using Neural Networks and Fuzzy Logic," in 2007 IEEE International Symposium on Intelligent Signal Processing. IEEE, 2007, pp. 1–6.
10. Navidi N., Landry R.J., Cheng J., Gingras D. (2016), A New Technique for Integrating MEMS-Based Low-Cost IMU and GPS in Vehicular Navigation, Journal of Sensors, Vol 2016.
11. Xing H., Hou B., Lin Z., Guo M. (2017), Modeling and Compensation of Random Drift of MEMS Gyroscopes Based on Least Squares Support Vector Machine Optimized by Chaotic Particle Swarm Optimization, Sensors, 17(10).
12. Minsky M.L. and Papert S.A. (1988), Perceptrons: Expanded Edition. MIT Press, Cambridge, MA, USA.
13. Mirowski P. and Lecun Y. (2009), Dynamic Factor Graphs for Time Series Modelling, In Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD, Vol. 5782 LNAI, pp. 128-143.
14. Gu J., Yang X., De Mello S., Kautz J. (2017), Dynamic Facial Analysis: from Bayesian Filtering to Recurrent Neural Network, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1531-1540.
15. Krishnan R.G., Shalit U., Sontag D. (2017), Structured Inference Networks for Nonlinear State Space Models, Thirty-First AAAI Conference on Artificial Intelligence.
16. Goodfellow I., Bengio I., Courville A. (2016), Deep Learning, the MIT Press.
17. Hochreiter S. and Schmidhuber J. (1997), Long Short-Term Memory, Neural Comput. 9, 8, pp.1735-1780.
18. Geiger A., Lenz P., Stiller C., and Urtasun R. (2013), Vision meets Robotics: The KITTI Dataset, International Journal of Robotics Research (IJRR).
19. Hosseinyalamdary S. and Balazadegan Y. (2017), Error Modeling of Reduced IMU using Recurrent Neural Network, Indoor Positioning and Indoor Navigation (IPIN), Sapporo, Japan.