Journal of Uncertainty Analysis and Applications
a SpringerOpen Journal

RESEARCH Open Access

Computational modeling for parallel grid-based recursive Bayesian estimation: parallel computation using graphics processing unit

Xianqiao Tong[1*], Tomonari Furukawa[1] and Hugh Durrant-Whyte[2]

*Correspondence: txq1986@vt.edu
[1] Department of Mechanical Engineering, Virginia Tech, 800 Drillfield Dr, Blacksburg, VA 24061, USA
Full list of author information is available at the end of the article

Abstract

This paper presents the performance modeling of the real-time grid-based recursive Bayesian estimation (RBE), particularly the parallel computation using graphics processing unit (GPU). The proposed modeling formulates data transmission between the central processing unit (CPU) and the GPU as well as floating point operations to be carried out in each CPU and GPU necessary for one iteration of the real-time grid-based RBE. Given the specifications of the computer hardware, the proposed modeling can thus estimate the total amount of time cost for performing the grid-based RBE in a real-time environment. A new prediction formulation, which adopted separable convolution, is proposed to further accelerate the real-time grid-based RBE. The performance of the proposed modeling was investigated, and parametric studies have first demonstrated its validity in various conditions by showing that the average error of estimation in computational performance stays below 6% to 7%. Utilizing the prediction with separable convolution, the grid-based RBE has also been found to perform within 1 ms, although the size of the problem was relatively large.

Keywords: RBE; Bayesian; GPU; Real-time; Grid-base; Parallel

Introduction

Recursive Bayesian estimation (RBE) allows the estimation of belief of a dynamically moving target by updating the belief both in time and observation [1]. There are two fundamental processes for the RBE: prediction process and correction process. The prediction process updates the belief by the motion model of the target, whereas the correction process updates the belief through the current observation. If the target is observable, the accuracy of the RBE can be maintained by the correction process using the valid observations. When the target is not observable, the accuracy of the RBE heavily relies on the prediction process and the error accumulates due to the lack of the valid observation for the correction process. In order for an accurate estimation, the RBE has to be performed fast enough to catch the motion of the target with a well-defined target motion model, which requires a good synchronization between its discrete representation and the RBE. Recent years, as a result, have seen many real-time enhanced RBE techniques that help improve the speed of the RBE.

One of such techniques is the modified ensemble Kalman filter (EnKF). The EnKF allows non-Gaussian estimation by minimizing a cost function defined by a non-Gaussian observation error with a pre-conditioned conjugate gradient method [2]. Langevin-Markov Chain Monte Carlo (MCMC) method, which represents the non-Gaussian belief by sampling it using a Markov chain and Langevin equation, could be a non-Gaussian RBE technique [3]. Another sampling method is the interactive particle filter (IPF), which is able to flexibly mitigate the belief space complexity [4]. An ensemble Kalman-particle predictor-corrector filter is a hybrid method that combines the advantages of EnKF and IPF and is able to effectively deal with high-dimensional non-Gaussian problems [5]. A tree-based estimator approximates the posterior belief distribution at multiple resolutions to be effective for high-dimensional problems [6], whereas maximum likelihood state estimation method could also achieve non-Gaussian RBE [7] by using a finite Gaussian mixture model.

Grid-based RBE technique is able to maintain a good accuracy for the belief since the entire target space is spatially discretized [8]. The good accuracy is obtained by the subtle discretization of the target space but leads to an inefficient computation at the same time. Furukawa et al. [9,10] refined the grid-based RBE by developing a more general element-based RBE. The generalized element can help accurately represent the arbitrary target space with only the small number of elements compared with the grid-based RBE so as to reduce the computation of the RBE. Lavis et al. proposed an enhanced grid-based RBE that allows the update of not only the belief but also the target space [11]. Because of the dynamic adjustment of the target space, the computation of the RBE is additionally reduced. Further, the parallel grid-based RBE has been proposed, and it significantly accelerated the computation of the RBE and made its real-time implementation possible by utilizing the GPU's strong parallel computational capability [12]. Despite that these efforts successfully reduce the computation of the RBE to achieve the fast RBE, the accuracy of the RBE is not well kept when the prediction process dominates the RBE during the no-observation period. The time cost of one iteration of the RBE becomes critical for overcoming this issue because that only if it matches the time increment of the discrete target motion model, the RBE can maintain the accuracy during the no-observation period.

This paper presents a performance modeling for the parallel grid-based RBE, particularly the parallel computation using the GPU, and it is able to determine the time cost of one iteration of the RBE. The proposed modeling formulates the total amount of data transmission between the CPU and the GPU and the total number of floating point operations to be carried out in each CPU and GPU necessary for one iteration of the parallel grid-based RBE. Given the specifications of the computer hardware, it is thus possible to estimate the time cost for one iteration of the parallel grid-based RBE. In order to perform the parallel grid-based RBE at maximum speed, the proposed modeling also reformulates and implements the prediction process with separable convolution.

The paper is organized as follows. The following section reviews the recursive Bayesian estimation as well as the parallel grid-based RBE. Section presents the proposed reformulation of the prediction process for the parallel grid-based RBE and its computational performance modeling. Section demonstrates the validation and efficacy of the proposed modeling through numerical examples, and the Conclusion and future work are summarized in the final section.

Parallel grid-based RBE

Problem statement

The motion of an object, o, is deterministically given by the following equation:

$$\dot{\mathrm{x}}^o = \mathbf{f}^o(\mathrm{x}^o, \mathbf{u}^o, \mathrm{w}^o, t), \tag{1}$$

where x^o represents the state of the object, u^o represents the object control input, w^o represents the system noise, which includes environmental influences on the target, and t represents the time. In general, the state of the object describes its two-dimensional location but may also include other variables such as velocity. Let the time interval between the consecutive time steps be defined as Δt. By integrating Equation (1), the state of the object at the time step k is given by

$$\mathrm{x}_k^o = \mathrm{x}_{k-1}^o + \int_{t_{k-1}}^{t_{k-1}+\Delta t} \mathbf{f}^o(\mathbf{x}^o, \mathbf{u}^o, \mathbf{w}^o, t)dt, \tag{2}$$

where t_{k-1} is the time which corresponds to the time step $k-1$.

Recursive Bayesian estimation

Prediction

The prediction process starts with the numerical implementation of the object motion model defined in Equation (2). For simplicity, the numerical integration is carried out by Riemann left sum algorithm. By dividing the time interval Δt between the consecutive time steps into n subintervals. The state of the object at the time step k is given by

$$\mathrm{x}_k^o = \mathrm{x}_{k-1}^o + \sum_{i=0}^{n-1} \mathbf{f}^o\left(\mathbf{x}^o, \mathbf{u}^o, \mathbf{w}^o, t_{k-1} + i\frac{\Delta t}{n}\right)\frac{\Delta t}{n}. \tag{3}$$

Let a sequence of the observations of the object from time step 1 to $k-1$ be defined as ${}^s\tilde{\mathbf{z}}_{1:k-1} \equiv \{{}^s\tilde{\mathbf{z}}_i | \forall i \in \{1, \ldots, k-1\}\}$. Notice here that $\tilde{(\cdot)}$ represents an instance of variable $(\cdot)$. The prediction process computes the belief of the current state $p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k-1})$ from the belief in the previous time step $p(\mathbf{x}_{k-1}^o | {}^s\tilde{\mathbf{z}}_{1:k-1})$. The prediction is iteratively carried out by Chapman-Kolmogorov equation and given by

$$\begin{aligned} &p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = \int_{\mathcal{X}^o} p(\mathbf{x}_k^o | \mathbf{x}_{k-1}^o) p(\mathbf{x}_{k-1}^o | {}^s\tilde{\mathbf{z}}_{1:k-1}) d\mathbf{x}_{k-1}^o, \end{aligned} \tag{4}$$

where $p(\mathbf{x}_k^o | \mathbf{x}_{k-1}^o)$ is the probabilistic representation of the object motion model defined in Equation (3), which maps the probability of transition from the previous state $\mathbf{x}_{k-1}^o$ to the current state $\mathbf{x}_k^o$. The prediction process at $k=1$ is carried out by letting $p(\mathbf{x}_{k-1}^o | {}^s\tilde{\mathbf{z}}_{1:k-1}) = p(\tilde{\mathbf{x}}_0^o)$, where $p(\tilde{\mathbf{x}}_0^o)$ is defined as a prior belief of the object in terms of the probability density function. Equation (4) indicates that the performance of the prediction process relies on the object motion model $p(\mathbf{x}_k^o | \mathbf{x}_{k-1}^o)$. Due to the fact that the object motion model is usually non-Gaussian when only prediction process applies to the RBE, the belief could eventually become heavily non-Gaussian.

Correction

The correction process is associated with the definition of the observation model. Let the probability of detection (PoD) be $0 \leq P_{\mathrm{d}}(\mathbf{x}_k^o) \leq 1$ as a reliable measure for detecting the

object in terms of the object state. Observable region ${}^s\mathcal{X}_k^o$ is defined as

$$ {}^s\mathcal{X}_k^o = \{\mathbf{x}_k^o | 0 < P_{\mathrm{d}}(\mathbf{x}_k^o) \leq 1\}. \tag{5} $$

The observation ${}^s\mathbf{z}_k$ at the time step k is given by

$$ {}^s\mathbf{z}_k = \begin{cases} \mathbf{h}^s(\mathbf{x}_k^o, \mathbf{v}_k^s) & \mathbf{x}_k^o \in {}^s\mathcal{X}_k^o \\ \varnothing & \mathbf{x}_k^o \notin {}^s\mathcal{X}_k^o, \end{cases} \tag{6} $$

where $\mathbf{v}_k^s$ represents the observation noise at the time step k, and $\varnothing$ represents an empty element, indicating that the observation contained no information on the object or that the target is unobservable when it is not within the observable region.

The correction process then computes the belief $p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k})$ given the corresponding observations up to the previous time step $p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k-1})$ and a new observation ${}^s\tilde{\mathbf{z}}_k$. The equation is derived by applying formulas for marginal distribution and conditional independence and given by

$$ \begin{aligned} &p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k}) \\ &\quad = \frac{l(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k) p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k-1})}{\int_{\mathcal{X}^o} l(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k) p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_{1:k-1}) d\mathbf{x}_k^o}, \end{aligned} \tag{7} $$

where $l(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k)$ represents the observation likelihood of $\mathbf{x}_k^o$. The observation likelihood is defined with reference to the PoD and is given by

$$ l(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k) = \begin{cases} p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k) & {}^s\tilde{\mathbf{z}}_k \in {}^s\mathcal{X}_k^o \\ 1 - P_{\mathrm{d}}(\mathbf{x}_k^o) & {}^s\tilde{\mathbf{z}}_k \notin {}^s\mathcal{X}_k^o, \end{cases} \tag{8} $$

where $p(\mathbf{x}_k^o | {}^s\tilde{\mathbf{z}}_k)$ is the probabilistic representation of the observation model defined in Equation (6). When the object is within the observable region, a positive observation is obtained and the observation likelihood is a probability density function given the current of the object observation. When the object is out of the observable region, the negative observation is defined with respect to the PoD as the observation likelihood. Due to the fact that the observation likelihood of the negative observation is non-Gaussian, when the negative observation occurs in the RBE, the object belief would immediately become heavily non-Gaussian.

Parallel grid-based RBE

Representation of target space and belief

The grid-based RBE achieves non-Gaussian belief estimation by first representing the arbitrary target space $\mathcal{X}^t$ in terms of a set of grid cells by constructing a rectangular space $\mathcal{X}^r$ that covers the target space. For simplicity, let us consider a two-dimensional target space, and it is represented as $\mathbf{m}^t = [x^t, y^t] \in \mathcal{X}^t$. The creation of a rectangular space $\mathcal{X}^r$ is achieved then by defining the minimum and maximum values of the target space

$$ \begin{aligned} x_{\min}^t = \min\{x^t\}, x_{\max}^t = \max\{x^t\} \\ y_{\min}^t = \min\{y^t\}, y_{\max}^t = \max\{y^t\} \end{aligned} $$

and subsequently creating a rectangular space as $\mathcal{X}^r = \{\mathbf{m} | \forall x \in [x_{\min}^t, x_{\max}^t], \forall y \in [y_{\min}^t, y_{\max}^t]\} \supseteq \mathcal{X}^t$, where $\mathbf{m} = [x, y]$. The grid space is further introduced by discretizing the rectangular space by n_x and n_y grid cells in two directions, respectively. The dimensions of a grid cell are defined as $\Delta x^r = (x_{\max}^t - x_{\min}^t)/n_x$ and $\Delta y^r = (y_{\max}^t - y_{\min}^t)/n_y$.

This results in introducing the center of each grid cell as

$$\bar{\mathbf{m}}^r_{i_x,i_y} = [\bar{x}^r_{i_x}, \bar{y}^r_{i_y}] = [\,(i_x - 0.5)\Delta x^r + x^t_{\min}, (i_y - 0.5)\Delta y^r + y^t_{\min}\,], \tag{9}$$

where $\forall i_x \in \{1, \ldots, n_x\}$ and $\forall i_y \in \{1, \ldots, n_y\}$. Each grid cell is defined as

$$\mathcal{X}^r_{i_x,i_y} = \{\mathbf{m} || x - \bar{x}^r_{i_x}| < \frac{1}{2}\Delta x^r, |y - \bar{y}^r_{i_y}| < \frac{1}{2}\Delta y^r\}. \tag{10}$$

Note that $\bigcup_{i_x=1}^{n_x}\bigcup_{i_y=1}^{n_y}\mathcal{X}^r_{i_x,i_y} = \mathcal{X}^r$ and $\bigcap_{i_x=1}^{n_x}\bigcap_{i_y=1}^{n_y}\mathcal{X}^r_{i_x,i_y} = \varnothing$. Finally, the selection of grid cells that represent the target space is performed by selecting a grid cell when its center is located in the target space, $\mathcal{X}^r_{i_x,i_y} \subset \mathcal{X}^t$ if $\bar{\mathbf{x}}^r_{i_x,i_y} \in \mathcal{X}^t$. The approximate target space derived by the processes described above is $\mathcal{X}^t \approx \{\mathcal{X}^r_1, \mathcal{X}^r_2, \ldots, \mathcal{X}^r_{n_g}\}$, where n_g is the number of grid cells approximating the target space.

The belief is usually represented by a probability density function over the target space. Similar to the discretization of the target space, the belief could also be represented discretely by grid cells. The position of each grid cell can be described in the two-dimensional integer space as $[i_x, i_y]$, where $i_x \in 1, \ldots, n_x$ and $i_y \in 1, \ldots, n_y$. With the integer representation, the belief at the grid cell $[i_x, i_y]$ can be represented as $p^{i_x,i_y}(\cdot)$.

Prediction

The prediction process requires the numerical evaluation of Equation (4). Given the belief of the previous state $p^{i_x,i_y}(\mathbf{x}^t_k|{}^s\tilde{\mathbf{z}}_{1:k-1})$ at the grid cell $[i_x, i_y]$ and the target motion model $p^{I_x,I_y}(\mathbf{x}^t_k|\mathbf{x}^t_{k-1})$ constructed in the matrix of size $I_x \times I_y$ as a convolution kernel, the predicted belief of the current state can be numerically computed as

$$\begin{aligned} &p^{i_x,i_y}(\mathbf{x}^t_k|{}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = p^{i_x,i_y}(\mathbf{x}^t_{k-1}|{}^s\tilde{\mathbf{z}}_{1:k-1}) \otimes p^{I_x,I_y}(\mathbf{x}^t_k|\mathbf{x}^t_{k-1}), \end{aligned} \tag{11}$$

where $\otimes$ indicates the two-dimensional convolution of the belief of the previous state with the probabilistic target motion model. Therefore, the belief of the current state is given by

$$\begin{aligned} &p^{i_x,i_y}(\mathbf{x}^t_k|{}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = \sum_{\beta=1}^{I_y}\sum_{\alpha=1}^{I_x} p^{\alpha,\beta}(\mathbf{x}^t_k|\mathbf{x}^t_{k-1}) p^{i_x-\alpha+1,i_y-\beta+1}(\mathbf{x}^t_{k-1}|{}^s\tilde{\mathbf{z}}_{1:k-1}). \end{aligned} \tag{12}$$

The parallelization of the prediction process is straightforward. Since the prediction at each grid cell, given by Equation (12), can be performed independently, the parallelization of the prediction corresponds to the parallelization of the equation and achieves a parallel efficiency of 100% in an ideal environment. However, this equation also shows that the computation for the prediction process is largely dominated by the size of the convolution kernel. In order for real-time performance, it is important that the convolution kernel of an appropriate size, which needs to be big enough to capture the motion of the target as well as small enough to perform fast computation, is utilized.

Correction

The correction process corresponds to the numerical computation of Equation (7). Given the predicted belief $p(\mathbf{x}^t_k|{}^s\tilde{\mathbf{z}}_{1:k-1})$ and the new observation likelihood $l^{i_x,i_y}(\mathbf{x}^t_k|{}^s\tilde{\mathbf{z}}_k)$ at the

grid cell $[i_x, i_y]$, the corrected belief is computed by

$$p^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k}) = \frac{q^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})}{A_c \sum_{\alpha=1}^{n_g} q^{\alpha}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})}, \tag{13}$$

where A_c is the area of a grid cell, and

$$q^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k}) = l^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_k) p^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k-1}). \tag{14}$$

The parallelization of the correction process requires the breakdown of the process as it identifies which subprocesses are parallelizable. By observing the mathematical operations, the correction process can be broken down into three steps:

1. Calculate $q^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ by multiplying the predicted belief $p^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k-1})$ with the observation likelihood $l^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_k)$;
2. Sum $\sum_{\alpha=1}^{n_g} q^{\alpha}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ and multiply the sum by A_c;
3. Calculate $p^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ by dividing $q^{i_x,i_y}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ by $A_c \sum_{\alpha=1}^{n_g} q^{\alpha}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$.

The breakdown indicates that steps 1 and 3 are grid-wise sub-processes, which can be conducted independently. Therefore, for the correction process, steps 1 and 3 can be computed in parallel, whereas step 2 is not parallelizable.

Target state evaluation

In the parallel grid-based RBE, the state of the target is evaluated by Equation (2) in the integral form at each time step. For an accurate evaluation of the target state an appropriate choice of the time interval Δt is necessary. Given a specific computer hardware configuration, each iteration of the parallel grid-based RBE requires the certain amount of time Δt_c to perform the computation, including both the prediction and correction processes. In order to achieve an accurate evaluation of the target state, the time interval Δt needs to be chosen such that it matches the Δt_c. As shown in Figure 1, only when the Δt is identical with the Δt_c the evaluated target states could match the real target states. When the Δt is smaller or larger than the Δt_c, the evaluation of the target states fails and eventually leads to large accumulated errors. The Δt_c is determined by not only the parallel grid-based RBE itself but also its computational performance for the specific computer hardware configuration.

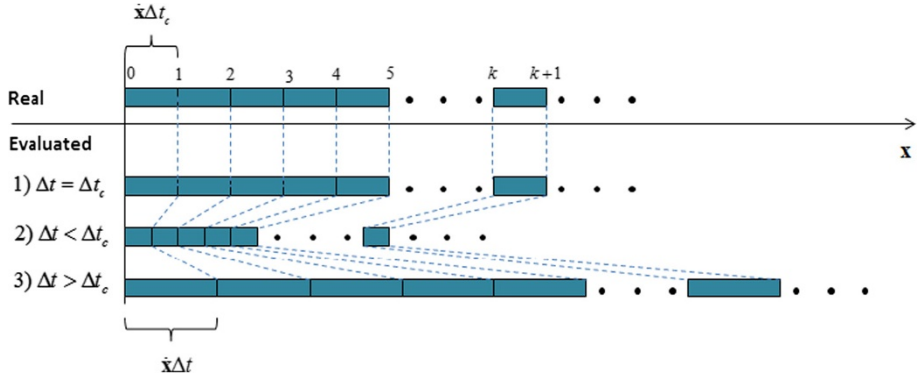


Figure 1 Target states evaluation.

Computational performance modeling

Acceleration of prediction process

Since the RBE designed with high frequency results in using the Markovian target motion model well approximated by a Gaussian probability density, the proposed modeling first reformulates the prediction process with the Gaussian assumption as a pre-process and accelerates the parallel grid-based RBE to achieve the maximum performance. With the Gaussian assumption, the convolution kernel in the matrix of size $I_x \times I_y$ can be separated into two vector kernels in the name of separable convolution: a column kernel of length I_x and a row kernel of length I_y. Therefore, the target motion model matrix is separated as

$$p^{I_x,I_y}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) = {}^c p^{I_x}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)\,{}^r p^{I_y}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t), \tag{15}$$

where ${}^c p^{I_x}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)$ and ${}^r p^{I_y}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)$ are the column kernel and row kernel, respectively, with the size of a vector of $I_x + I_y$. Substituting Equation (15) into Equation (11), the predicted belief of the current state can be computed as

$$\begin{aligned} &p^{i_x,i_y}(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = p^{i_x,i_y}(\mathbf{x}_{k-1}^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \otimes {}^c p^{I_x}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) \otimes {}^r p^{I_y}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t), \end{aligned} \tag{16}$$

which means that the prediction process can be broken down into two steps:

$$\begin{aligned} &u^{i_x,i_y}(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = p^{i_x,i_y}(\mathbf{x}_{k-1}^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \otimes {}^c p^{I_x}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) \\ &\quad = \sum_{\alpha=1}^{I_x} {}^c p^{\alpha}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) p^{i_x-\alpha+1,i_y}(\mathbf{x}_{k-1}^t|{}^s\tilde{\mathbf{z}}_{1:k-1}), \end{aligned} \tag{17}$$

and

$$\begin{aligned} &p^{i_x,i_y}(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \\ &\quad = u^{i_x,i_y}(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}) \otimes {}^r p^{I_y}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) \\ &\quad = \sum_{\beta=1}^{I_y} {}^r p^{\beta}(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t) u^{i_x,i_y-\beta+1}(\mathbf{x}_{k-1}^t|{}^s\tilde{\mathbf{z}}_{1:k-1}). \end{aligned} \tag{18}$$

These equations show that the prediction process at each grid cell is carried out by performing two one-dimensional convolutions, each in horizontal and vertical directions instead of the original one two-dimensional convolution while remaining complete parallelizability. For Equation (17), the number of floating point operations for each grid cell is seen $2I_x$ since I_x times of one multiplication and one summation are necessary, whereas the number of floating point operations for Equation (18) is $2I_y$ via the similar observation. Having a total of n_g grid cells, the total number of floating point operations for the prediction process is thus given by

$$N_{\mathrm{p}} = 2n_g I_x + I_y. \tag{19}$$

This is considerably small compared to that of the original formulation which is derived as $2n_g I_x I_y$ via Equation (12) since $I_x + I_y \ll I_x I_y$ for an appropriate prediction process.

Parallel computation using GPU

Following Equations (16) and (13) for the prediction and correction process, respectively, Figure 2 shows the schematic diagram of the proposed accelerated parallel grid-based RBE using GPU. For efficiency, the GPU stores the entire data for RBE in the global memory and performs RBE using local memories. As a result, the data transmission between the CPU's memory and the GPU's local memories is carried out via the GPU's global memory, and all the parallelizable floating point operations are executed using the local memories. For the prediction process, the data to be transmitted from the CPU's memory to the GPU's local memories are the previous belief $p(\mathbf{x}_{k-1}^t|^s\tilde{\mathbf{z}}_{1:k-1})$ and the target motion model $p(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)$. Since the predicted belief is in the local memories, the correction needs only the observation likelihood to be initially transmitted in addition. After performing the multiplication of $p(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k-1})$ and the observation likelihood $l(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_k)$ using GPU's local memories, the result $q(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ is transmitted to the CPU's memory to calculate the sum $A_c\sum_{\alpha=1}^{n_g} q^\alpha(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$. The sum is then transmitted back to the GPU's local memories to perform divisions in parallel and update the belief $p(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$. Finally, the belief is transmitted back to the CPU's memory for the next iteration of the accelerated parallel grid-based RBE.

Modeling of computational performance

The computational performance of the accelerated parallel grid-based RBE using GPU is determined not only by the performance of the CPU but also by the performance of the GPU and that of data transmission. As a result, the time cost of one iteration of the accelerated parallel grid-based RBE is given by

$$\Delta t_c = \Delta t_{\text{trans}} + \Delta t_G + \Delta t_C, \tag{20}$$

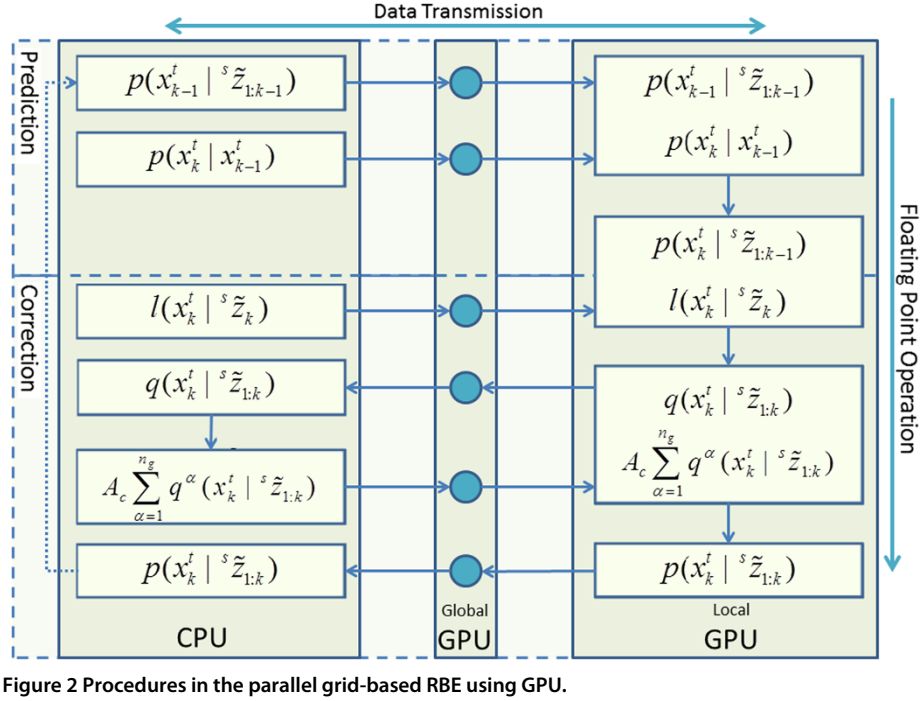


Figure 2 Procedures in the parallel grid-based RBE using GPU.

where Δt_{trans} represents the data transmission time cost between the CPU's memory and the GPU's global memory as well as that between the local and the global memory inside the GPU, Δt_{G} represents the time cost of the parallel computation performed on the GPU, and Δt_{C} represents the time cost of the computation performed on the CPU.

Data transmission

In order to determine the data transmission time cost Δt_{trans} for one iteration of the accelerated parallel grid-based RBE, the data transmitted among the CPU's memory, GPU's global memory, and GPU's local memory need to be evaluated in both the prediction and correction processes. Let the amount of data transmitted in the unit of bytes be defined as

$$A = PN, \tag{21}$$

where P is the precision of the numerical representation, and N is defined as the number of data transmitted. Since the precision is usually constant, the amount of data transmitted could be derived in terms of the number of data transmitted. The numbers of data of the belief and the target motion model for the prediction process are n_g and $I_x + I_y$, respectively. The same numbers of data, n_g and $I_x + I_y$, are transmitted to the GPU's local memory to perform parallel calculation. In the correction process, the number of data of the likelihood to be transmitted from the CPU's memory to the GPU's local memory through the GPU's global memory is n_g, whereas the number of data of the result $q(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$ to be transmitted from the GPU's local memory to the CPU's memory through the GPU's global memory is similarly n_g. The number of data of the sum, $A_{\text{c}} \sum_{\alpha=1}^{n_g} q^{\alpha}(\mathbf{x}_k^t|^s\tilde{\mathbf{z}}_{1:k})$, to be then transmitted to the GPU's local memory to perform parallel divisions is 1, and finally, the number of data to be transmitted back to the CPU's memory for the next RBE is n_g.

By observing the data transmission for one iteration of the accelerated parallel grid-based RBE, the total number of data transmitted from the CPU's memory to the GPU's global memory is given by

$$\begin{aligned} N_{\text{CG}} &= n_g + I_x + I_y + 1 + n_g \\ &= 2n_g + I_x + I_y + 1, \end{aligned} \tag{22}$$

and all the data are transmitted continuously from the GPU's global memory to the GPU's local memory

$$N_{\text{GL}} = N_{\text{CG}} = 2n_g + I_x + I_y + 1. \tag{23}$$

The total number of data transmitted from the GPU's local memory to the GPU's global memory is

$$N_{\text{LG}} = n_g + n_g = 2n_g, \tag{24}$$

and that from the GPU's global memory to the CPU's memory similarly becomes

$$N_{\text{GC}} = N_{\text{LG}} = 2n_g. \tag{25}$$

The data transmission time cost Δt_{trans} for one iteration of the accelerated parallel grid-based RBE is given by

$$\Delta t_{\text{trans}} = P\left(\frac{N_{\text{CG}}}{B_{\text{CG}}} + \frac{N_{\text{GC}}}{B_{\text{GC}}} + \frac{N_{\text{GG}}}{B_{\text{GG}}}\right), \tag{26}$$

where N_{CG} and B_{CG} are the total number of data transmitted and the copy bandwidth with the unit of bytes per second from the CPU's memory to the GPU's global memory, respectively, N_{GC} and B_{GC} are those from the GPU's global memory to the CPU's memory, respectively, and N_{GG} and B_{GG} represent those between the GPU's global memory and the GPU's local memory. Due to the fact that the copy bandwidth from the GPU's global memory to the GPU's local memory and the one in opposite direction are the same, the number of data transmitted inside the GPU is given by

$$N_{GG} = N_{GL} + N_{LG} = 4n_g + I_x + I_y + 1. \tag{27}$$

Substitute Equations (22), (25), and (27) into Equation (26), the data transmission time cost for one iteration of the accelerated parallel grid-based RBE is given by

$$\Delta t_{trans} = P\left(\frac{2n_g + I_x + I_y + 1}{B_{CG}} + \frac{2n_g}{B_{GC}} + \frac{4n_g + I_x + I_y + 1}{B_{GG}}\right). \tag{28}$$

It is to be noted here that these parameters of copy bandwidths are inherent for a specific computer hardware configuration and can be determined experimentally.

Floating point operations

In order to determine the GPU computation time cost Δt_G and CPU computation time cost Δt_C for one iteration of the accelerated parallel grid-based RBE, the number of floating point operations performed on both CPU and GPU needs to be evaluated. The number of floating point operations performed on the GPU for the prediction process is seen $2n_g(I_x + I_y)$ as the Equation (19) indicated. The number of floating point operations performed on the GPU for the correction process is identified as $2n_g$ in total since n_g parallel multiplications and n_g parallel divisions are performed for steps 1 and 3 in Subsection 2, respectively. Meanwhile, the number of floating point operations performed on the CPU is n_g by n_g summations in step 2 of the Subsection 2. As a consequence, the total number of floating point operations performed on the GPU and the CPU for one iteration of the accelerated parallel grid-based RBE is given, respectively, by

$$N_G = 2n_g(I_x + I_y) + 2n_g = 2n_g(I_x + I_y + 1),$$

$$N_C = n_g.$$

The GPU computation time cost for one iteration of the accelerated parallel grid-based RBE is given by

$$\Delta t_G = \frac{N_G}{V_G}, \tag{29}$$

where N_G is the number of floating point operations performed on the GPU, and V_G is the computational rate of GPU with the unit of FLOPS. Substituting Equation (29) into Equation (29), the GPU computation time cost is given by

$$\Delta t_G = 2n_g \frac{I_x + I_y + 1}{V_G}. \tag{30}$$

Similarly, the CPU computation time cost for one iteration of the accelerated parallel grid-based RBE is given by

$$\Delta t_C = \frac{N_C}{V_C}, \tag{31}$$

where N_C represents the number of floating point operations performed on the CPU, and V_C is the computational rate of CPU with the unit of FLOPS. In the same way, by substituting Equation (29) into Equation (31), the CPU computation time cost is given by

$$\Delta t_C = \frac{n_g}{V_C}. \tag{32}$$

It is to be noted here that the computational rates, V_G and V_C, are also inherent for a specific CPU and GPU configuration and can be determined experimentally.

Experimental validation

Table 1 shows the setup specifications which have been available for the validation and other investigations. Setup 1 is the fastest in both CPU and GPU, whereas setup 3 is the slowest. This section firstly shows the improvement of the parallel grid-based RBE using GPU by adopting the separable convolution in the prediction process with the specification listed in setup 1. Moreover, the proposed computational modeling for the parallel grid-based RBE is validated via setups 1 to 3. In the end, a simulated target searching task is introduced to further evaluate the efficacy of the proposed modeling.

Improvement in prediction process

The efficiency of the prediction process accelerated by separable convolution was evaluated with a problem having a fixed grid space size of $1,000 \times 1,000$ and varying the convolution kernel size from 1 to 50 on the computer setup 1. The result of the time cost by GPU is shown in Figure 3 together with the corresponding result by the original prediction. Even when the convolution kernel size is 50, the accelerated prediction is seen to require the time cost of only 1 ms. Its superiority can also be understood by comparing it to the original prediction, which needs the time cost 25 times as much as that of the accelerated prediction process when the convolution kernel size is 50.

Validation

This set of tests was aimed at validating the proposed modeling of computer performance by estimating the total iteration time cost Δt of the parallel grid-based RBE using GPU and comparing it with the actual iteration time cost experimentally measured in three different computer setups. Each component, Δt_{trans}, Δt_G, or Δt_C, is also compared with the actual performance, respectively. All the time cost results are measured by averaging the time cost of $10,000$ iterations. Needless to say, the convolution kernel size $I_x + I_y$ and grid space size n_g are the two major factors in the proposed modeling. Two tests were thus conducted by each, changing the convolution kernel size and the grid space size.

Table 1 Test computer system specifications

Setup	Processor	Memory (GB)	GPU
1	Intel Dual-Core, 2.70 GHz	4.0	Nvidia GeForce GT220
2	Intel Dual-Core, 2.40 GHz	4.0	Nvidia GeForce GT320M
3	Intel Dual-Core, 2.40 GHz	4.0	Nvidia GeForce GS8400

Nvidia GeForce (Santa Clara, CA, USA), Intel (Santa Clara, CA, USA).

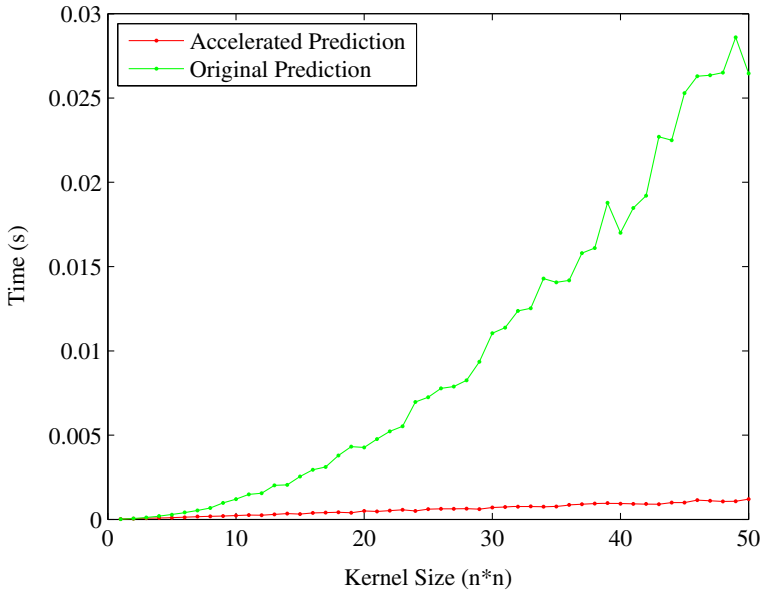


Figure 3 GPU calculation time cost.

Test 1

Test 1 was performed by fixing the grid space size of the parallel grid-based RBE to $1,000 \times 1,000$ and varying the convolution kernel size $I_x = I_y = i$ from 1 to 200. A convolution kernel size over 200 was not explored since it is unlikely that the target motion model requires such a large convolution kernel. The square convolution kernel was because of the insignificance in changing size in both x and y directions, and this additionally allows visualization of results in two-dimensional space.

The results of all the components of the time cost for the three computer setups are shown in Figures 4, 5, and 6. Each solid line represents the estimated total and component time costs, whereas each solid dot line with the same color represents the corresponding actual performance. These figures primarily show that the total and component time costs estimated by the proposed modeling well match to the actual performance. Values listed in Table 2 also support this and indicate the effectiveness of the proposed modeling since the average and the maximum relative errors are below 7% and 12%, respectively. While the time cost of data transmission is seen to contribute most, it is also seen that the time cost by GPU increases the total time cost with increase in convolution kernel size particularly when the GPU is of low quality. It is thus important to use a high-performance GPU if fast RBE with large convolution kernel size is necessary.

Test 2

Test 2 was performed by fixing the convolution kernel size of the parallel grid-based RBE to 16×16 or 32×32 and varying grid space size $n_x = n_y = n$ from 100 to 1,000. These convolution kernel sizes often represent the target motion model with sufficient accuracy, and the grid space size $n = 1,000$, which creates $1,000,000$ grid cells, also provides good accuracy in many practical problems. Similarly to test 1, the square grid size enables two-dimensional visualization of results.

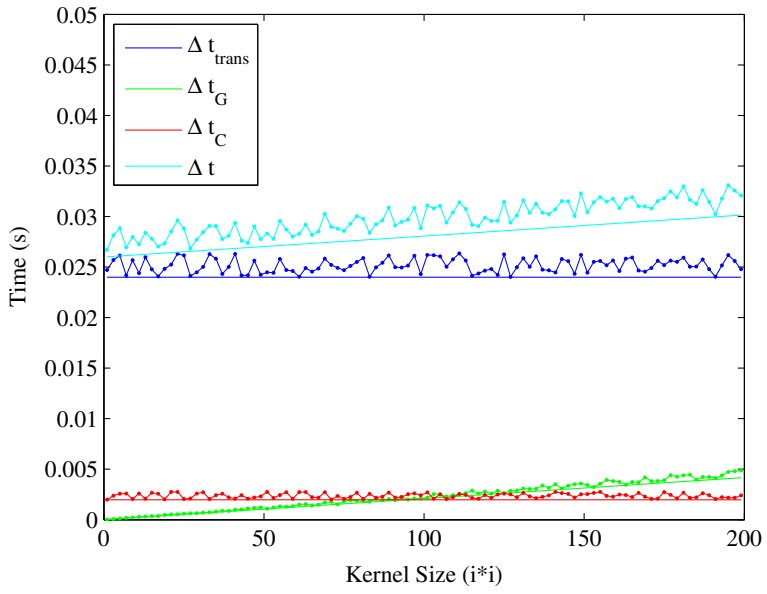


Figure 4 Time cost of all the components for setup 1 with fixed grid space.

The results for all the components of the time cost for the three computer setups are shown in Figures 7, 8, and 9, respectively. These figures firstly show that the proposed modeling is also able to well estimate the actual performance of the parallel grid-based RBE regardless of different grid space sizes. Similarly to test 1, Table 3 shows small average and maximum relative errors, which are below 6% and 11%, respectively. Secondly, from these results, it is seen that the total time cost is dominated by the time cost of data transmission particularly when the ratio of the grid space size to the convolution kernel

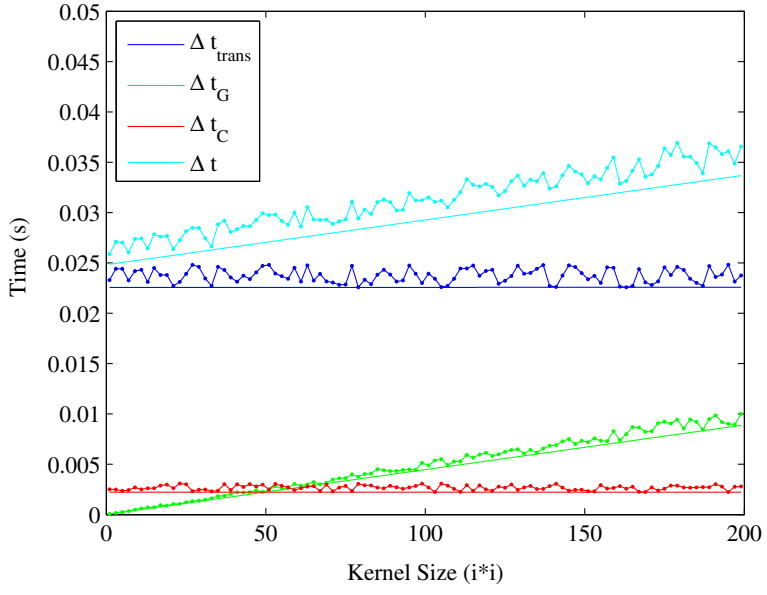


Figure 5 Time cost of all the components for setup 2 with fixed grid space.

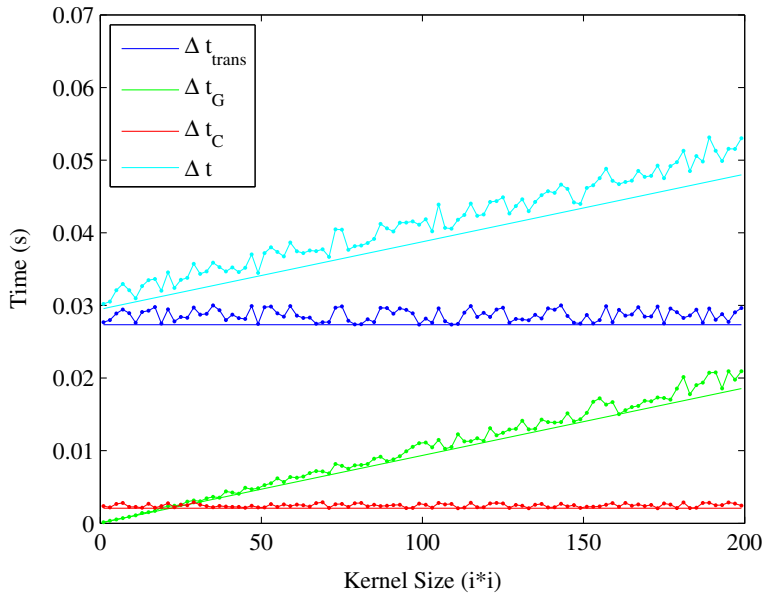


Figure 6 Time cost of all the components for setup 3 with fixed grid space.

size is large. Since the data transmission rate is determined by the quality of the memory, the utilization of a high-quality memory is the first priority for fast RBE.

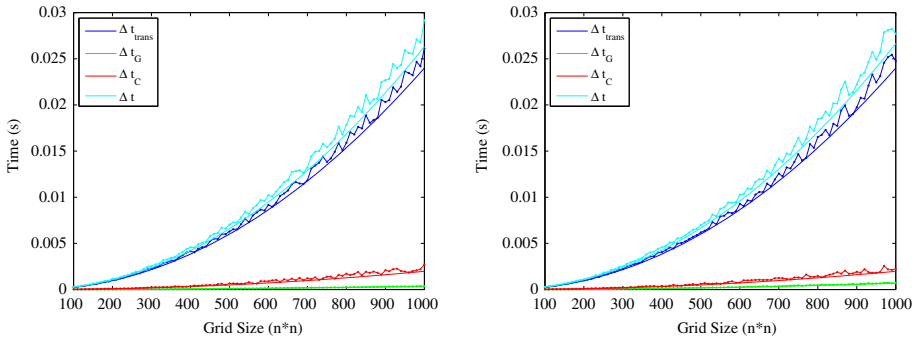
Simulated target searching task

The performance of the prediction process dominates the accuracy of the RBE when no valid observations are obtained. The aim of this test is to evaluate how well the proposed modeling help the prediction process keep the accuracy during the no observation period. A simplified target searching task is described in this subsection. The motion model of the simulated target is given by

$$\begin{aligned} x_{k+1}^t &= x_k^t + \Delta t \cdot v_k^t \cos \gamma_k^t \\ y_{k+1}^t &= y_k^t + \Delta t \cdot v_k^t \sin \gamma_k^t, \end{aligned} \tag{33}$$

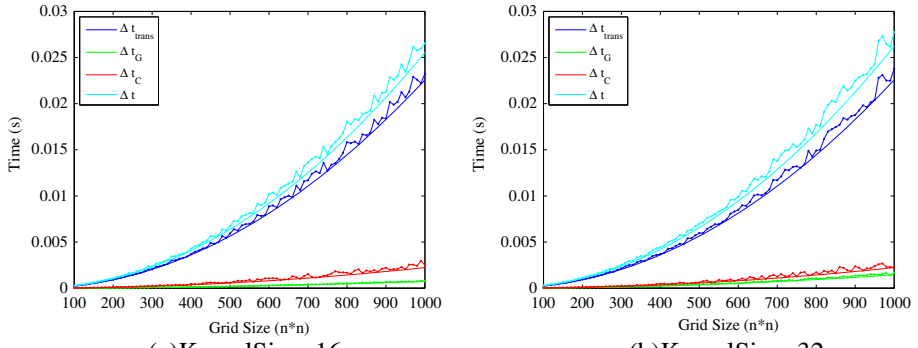
Table 2 Quantitative results for test 1

	Time cost	Setup		
		1	2	3
Average relative error	Δt_{trans}	1.159 ms	1.165 ms	1.305 ms
	Δt_{G}	0.216 ms	0.462 ms	0.856 ms
	Δt_{C}	0.402 ms	0.446 ms	0.382 ms
	Δt	1.777 ms	2.073 ms	2.543 ms
		(5.88%)	(6.55%)	(6.05%)
Maximum relative error	Δt_{trans}	2.351 ms	2.254 ms	2.670 ms
	Δt_{G}	0.716 ms	1.464 ms	3.259 ms
	Δt_{C}	0.779 ms	0.857 ms	0.818 ms
	Δt	3.228 ms	4.149 ms	6.081 ms
		(10.63%)	(11.24%)	(11.45%)



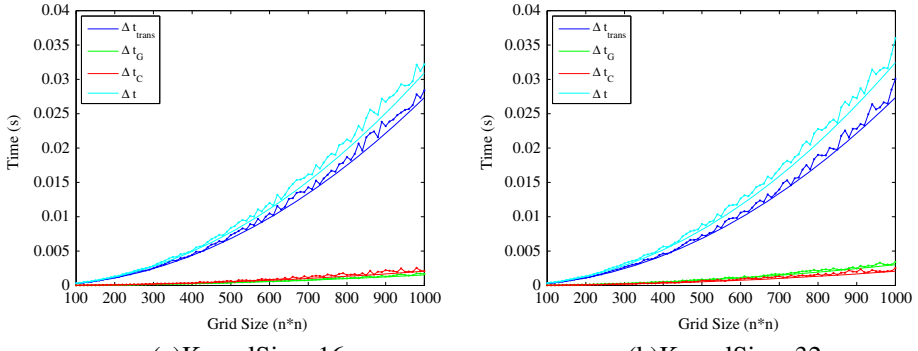
(a)KernelSize=16 (b)KernelSize=32

Figure 7 Time cost of all the components for setup 1 with fixed kernel.



(a)KernelSize=16 (b)KernelSize=32

Figure 8 Time cost of all the components for setup 2 with fixed kernel.



(a)KernelSize=16 (b)KernelSize=32

Figure 9 Time cost of all the components for setup 3 with fixed kernel.

Table 3 Quantitative results for test 2

	Total time cost	Setup		
		1	2	3
Average relative error	Δt	0.513 ms	0.530 ms	0.617 ms
		(5.59%)	(5.68%)	(5.90%)
Maximum relative error	Δt	2.140 ms	2.491 ms	2.835 ms
		(10.08%)	(10.64%)	(10.26%)

where v^t and γ^t are the velocity and direction of the target motion, respectively, each subject to a Gaussian noise, and Δt is the time increment. The prior belief on the target is also Gaussian. The autonomous sensor platforms are assumed to move on a horizontal plane and given by

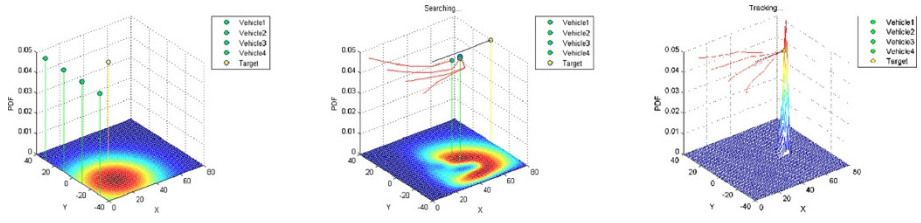
$$\begin{aligned} x_{k+1}^{s_i} &= x_k^{s_i} + \Delta t \cdot v_k^{s_i} \cos \gamma_k^{s_i} \\ y_{k+1}^{s_i} &= y_k^{s_i} + \Delta t \cdot v_k^{s_i} \sin \gamma_k^{s_i} \\ \theta_{k+1}^{s_i} &= \theta_k^{s_i} + \Delta t \cdot \alpha^{s_i} \gamma_k^{s_i}, \end{aligned} \tag{34}$$

where v^{s_i} and γ^{s_i} are the velocity and turn of the sensor platform (s_i) respectively, and α^{s_i} is a coefficient governing the rate of turn. The probability of detection $P_{\mathrm{d}}(\mathbf{x}_k^t|\mathbf{x}_k^{s_i})$ is given by a Gaussian distribution, whereas the likelihood $l(\mathbf{x}_k^t|^{s_i}\tilde{\mathbf{z}}_k^t, \tilde{\mathbf{x}}_k^{s_i})$ when the target is detected is given by a Gaussian distribution with variances proportional to the distance between the sensor platform s_i and the target. Table 4 shows the major parameters of this simulated target searching task. The convolution kernel constructed by the target motion model is represented by a 32×32 matrix, and the grid space size is set as $1,000 \times 1,000$. The computer specifications followed the setup 3 in the Table 1. With the proposed approach, the time increment Δt was chosen as 0.032 s, the time cost of one iteration of the RBE estimated by the proposed modeling. For the case without the proposed approach, the time increment Δt was chosen as 0.02 s randomly in order to show the comparison.

Figure 10 shows the initial and final states of four sensor platforms without and with proposed prediction reformulation, respectively. Without the proposed prediction improvement, all the sensor platforms lost the target, whereas all of them successfully found the target under the condition of utilizing the proposed prediction reformulation. The reason is that the proposed prediction process made the grid-based RBE to update the belief much faster than the original one, resulting in a reliable tracking on the moving target. The evaluation of the proposed modeling for this simulated search and rescue task

Table 4 Major parameters of the target searching task

	Parameter	Value
Sensor platform, s_i	Velocity $v_k^{s_i}$	0.05
	Turn coef. $\boldsymbol{\alpha}^{s_i}$	0.8
	PoD var.	[0.2, 0.2]
Target, t	Velocity v_k^t	N(0.1, 0.02)
	Direction γ_k^t	N(0rad, 0.7rad)
	Prior $[x_0^t, y_0^t]$	N([20, 25], diag{200, 200 })



(a) Initial state

(b) Final state (original approach)

(c) Final state (propose approach)

Figure 10 Search and rescue task.

is conducted, and the corresponding quantitative results were concluded in the Table 5. The result shows small average and maximum relative errors as well, which are below 7% and 10% respectively, and indicates that the proposed modeling is able to estimate the actual time cost for the grid-based RBE using GPU.

Conclusion and future work

The performance modeling for the real-time grid-based RBE, especially parallel computation using GPU, has been proposed to identify the best resolution of the RBE with given computer hardware. The modeling allows the estimation of time costs necessary within CPU and GPU and that of data transmission between CPU and GPU for the real-time grid-based RBE. In order to speed up the RBE, the prediction has been additionally reformulated with the separable convolution.

The proposed modeling was experimentally investigated by varying its major parameters. The result of the first test with varying convolution kernel size shows that the average error of the estimation by the proposed modeling stays below 7% regardless of the convolution kernel size and that a high-performance GPU is necessary if the convolution kernel size is large. In the second test with varying grid space size, it is found that the proposed modeling estimates within the average error of 6%, irrespective of the grid space size, and that a high-quality memory is necessary if fast RBE is required for large grid space. Utilizing prediction with separable convolution, the RBE has also been found to perform within 1 ms, although the size of the problem was relatively large.

The current study is still the first step for achieving high-fidelity RBE in a real-time environment. The project is further planned to utilize the best resolution of the RBE identified by the proposed modeling and investigate its efficacy.

Table 5 Quantitative results for simulated search and rescue task

	Total time	Sensor platform			
		1	2	3	4
Average relative error	Δt	0.618 ms	0.633 ms	0.626 ms	0.618 ms
		(5.78%)	(6.21%)	(5.82%)	(5.93%)
Maximum relative error	Δt	2.856 ms	2.823 ms	2.892 ms	2.854 ms
		(9.89%)	(9.56%)	(9.25%)	(9.68%)

Author details
[1]Department of Mechanical Engineering, Virginia Tech, 800 Drillfield Dr, Blacksburg, VA 24061, USA. [2]Australian Centre for Field Robotics (ACFR), Rose St, Sydney 2006, Australia.

Received: 29 August 2013 Accepted: 11 November 2013
Published: 16 December 2013

References

1. Tarantola, A: Inverse Problem Theory and Methods for Model Parameter Estimation. Society for Industrial and Applied Mathematics, Philadelphia (2005)
2. Harlim, J, Hunt, BR: A non-Gaussian ensemble filter for assimilating infrequent noisy observations. Tellus A. **59**, 225–237 (2007)
3. Apte, A, Hairer, M, Stuart, AM, Voss, J: Sampling the posterior: an approach to non-Gaussian data assimilation. Physica D. **230**, 50–64 (2007)
4. Doshi, P, Gmytrasiewicz, PJ: Monte Carlo sampling methods for approximating interactive POMDPs. J. Artif. Intell. Res. **34**, 297–337 (2009)
5. Mandel, J, Beezley, JD: An ensemble Kalman-particle predictor-corrector filter for non-Gaussian data assimilation. Comput. Sci. ICCS. **2009**, 470–478 (2009)
6. Stenger, B, Thayananthan, A, Torr, PHS, Cipolla, R: Filtering using a tree-based estimator. IEEE Int. Conf. Comput. Vis. **2**, 1063–1070 (2003)
7. Huang, D, Leung, H: Maximum likelihood state estimation of semi-Markovian switching system in non-Gaussian measurement noise. IEEE Trans. Aerosp. Electron. Syst. **46**, 133–146 (2010)
8. Bergman, N: Recursive Bayesian estimation navigation and tracking applications. PhD Dissertation, Linkopings University (1999)
9. Furukawa, T, Durrant-Whyte, HF, Lavis, B: The element-based method—theory and its application to Bayesian search and tracking. Paper presented at the IEEE/RSJ international conference on intelligent robots and systems, San Diego (29 Oct–2 Nov 2007)
10. Lavis, B, Furukawa, T: HyPE: Hybrid particle-element approach for recursive Bayesian searching and tracking. Proceedings of Robotic: Science and Systems IV, pp. 135–142. MIT Press, Zurich (2008)
11. Lavis, B, Furukawa, T, Durrant-Whyte, HF: Dynamic space reconfiguration for Bayesian search and tracking with moving targets. Auto. Robots. **24**(4), 387–399 (2008)
12. Furukawa, T, Lavis, B, Durrant-Whyte, HF: Parallel grid-based recursive Bayesian estimation using GPU for real-time autonomous navigation. Paper presented at the IEEE international conference on robotics and automation, Anchorage, AK, USA (3–7 May 2010)

doi:10.1186/2195-5468-1-15
Cite this article as: Tong *et al.*: **Computational modeling for parallel grid-based recursive Bayesian estimation: parallel computation using graphics processing unit.** *Journal of Uncertainty Analysis and Applications* 2013 **1**:15.