Optimization of mobile updates using Particle filter

Vesselin Tzvetkov
vesselin.tzvetkov@arcor.net
Arcor AG&Co KG, Alfred-Herrhausen-Allee 1, 65760 Eschborn, Germany

***Abstract*- The mobile usage of Internet is characterized by frequent changes of the access network and consequently, changes of the application's IP address or port. Intermediate NAT devices can exchange additionally the transport and network headers. Not using the current IP and port parameter leads to lost packets and service interruption. To overcome these problems, the applications send updates or keep-a-lives in regular basis, for example Dead-Peer-Detection in IKE. These messages inform the communication participants that the host is reachable. The main shortcoming is that the updates are performed at constant intervals regardless of the network properties. The problem oscillates in mobile environment where frequent network changes are expected. The result is wasted resources and long disconnection intervals. The key idea in this work is to set the update intervals proportional to the probability for network change. The probability density function is built using the past disconnections, thus the history is used to optimize the update intervals. Novel framework based on Particle filter is derived and simulated in this paper. The new method outperforms significantly the classical constant updates. Many protocols in mobile environment can profit from the new framework, like SIP, IKE, Routing protocols etc.**

***Index Terms*- update interval, keep-a-live, Particle filter, Sequential Monte Carlo, NAT, mobile environment**

I. Introduction

The Internet is becoming the dominating medium for modern communications. The classical broadcast, phone and data networks converge to services of IP network building the Next Generation Network (NGN). Staying "online" becomes vital not only for business, but also for private users. A consequence of portable devices and mobile lifestyles is mobile usage of Internet, thus enabling access to the Internet whilst on the move.

The IP and port parameter should be known to the communication participants, otherwise, the sent packet get lost i.e. the service is interrupted. An Influence on the application's the IP and port have: (1) The mobile usage of Internet leads to frequent changes of the access network. Every access network has assigned statically IP ranges. Changing the access network means change of the host's IP. (2) Significant issue is the widespread of NAT devices. Almost every broadband access to Internet involves NAT router. A host behind a NAT is not aware of it public IP and port since this values are manipulated by the intermediate device. Every change of the NAT binding at this intermediate device reflects in a change of the IP or port parameter of the application behind. (3) The multi-homed host are also a problem, i.e. host having multiple IP addresses. An example could be a smart phone connected to Internet with 3G and WiFi interface. The issue arises since the change of the outgoing interface reflects in change of the source IP address.

To eliminate these issues, the application must proactively detach a change of the connection status end-to-end. The status of the local interface is not sufficient for concluding if the peer is reachable. For these reasons, the applications involve update or keep-a-live procedures. The notation depends on the protocols. For example: Dead-Peer-Detection in IKE, Registration in SIP, binding update in Mobile IP etc. The principle is the same in all cases. The host sends a message and expects a response if the connection is active. By receiving an update message, the IP and UDP are updated and the host considered as reachable. The term "update" summaries the procedures in this paper.

The main drawback of the current update procedure is that it is executed in constant intervals. The values are commonly chosen by subjective experience, like 60 sec for SIP registration. The interval is not adapted to the changing network properties and mobile behavior of the node. In mobile environments, this leads to underperformance in respect of disconnection time by resources. The network can be overloaded by updates and in the same time, the host may suffer from frequent service lost.

II. Targets and objectivies

The key idea in this work is to set the update intervals proportional to the probability of disconnection. The updates must be frequent when there is high probability of IP or port change. On the contrarily, the updates must be rare when there is low probability of disconnection. The Figure 1 presents the principle. The time points of updates are shown at the x-axis and the Probability Density Function (PDF) at the y-axis. A practical example can be worker day cycle with high activity during the day and low at night.

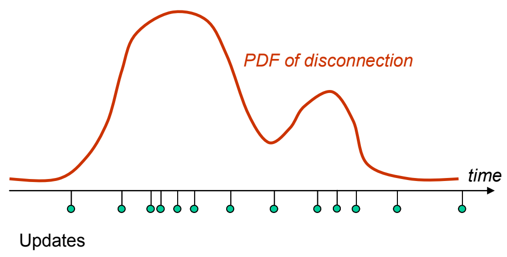

Figure 1 Updates dependent on the probability of disconnection.

The PDF is constructed using the history of disconnections. The PDF is dynamic depending on the network and mobile host properties. In this way, resources are saved and the disconnection reduced.

III. Difficulties

The well-developed classical estimation methods, like Particle filter, cannot be deployed in a straightforward way. They require numerical values delivered by measurement. The measurement consists commonly of signal with added noise. The target is to estimate the original signal.

The execution of update delivers Boolean result and not numerical values. The Boolean value gives the information if application is still reachable, thus if the IP address or port of has changed. The time point of change is unknown to the mobile host. The IP and port change can be due intermediate NAT, which do not notify the participants. There are not numerical measurements and the classical estimation method cannot be deployed directly. The time point of disconnection can be narrowed down to the update interval (time between the updates). For example: if the update interval is 300 seconds, then the disconnection could happen everywhere in the 300 sec.

Another issue is the dual purpose of the update. On the one hand, the update gives the information if the connection is still active. On the other hand, it informs the receiver about the current IP and UDP parameters. Expressed in classical terms, the measurement sets the signal in the same time. The update defines the interval, where the disconnection could happen.

Decreasing the update interval decreases the disconnection and improves the precision of knowing where the disconnection happened. Unfortunately, the updates consume network and host recourses. There is a clear trade off between the update intervals and the consumed resources. It can be compared to calling the information desk of the railway station and asking if the train has already passed. When the answers are only "yes or no", it becomes quite recourse consuming task to detach the arrival time, since the phone call costs money.

IV. Contributions

First of all, a model for the mobile updates is created, which allows the implementation of particle filter. Secondly, the author implements a new treatment of the degeneracy problem. Thirdly, an algorithm for particle moving is suggested in order to work with zero-knowledge systems. Furthermore, a practical contribution is the proof of concept through simulation showing the qualities of the new developed method and comparing them to constant update intervals.

V. Abstraction and Terminology

The update procedures, like keep-a-live or dead-peer-detection, are abstracted in order to develop a general model. The abstraction does not change the nature of the procedure.

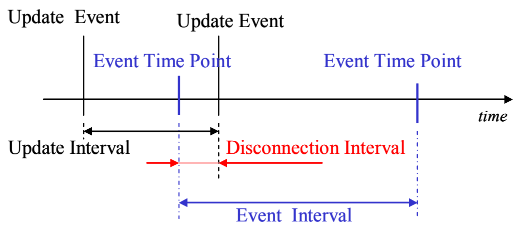

Figure 2 Abstraction

The term *move* denotes a change of application's IP addresses or port, thus the application become reachable under new parameters. It may, but must not, be related to physical movement. The time point of the host movement is denoted as *Event Time Point* (ETP), see Figure 2. The procedure of updating the IP and port is called *update*. The update procedure returns *true* when the host has not moved between the previous and current execution. The result *false* means the host has moved, thus IP or port has changed. The points, where the update procedure is executed, are called *Update Time Points* (UTP). The interval between two following UTPs is *Update Intervals* (UI). The *Event Intervals* (EI) is the interval between the two following ETPs.

The *Disconnection Interval* (DI) is the interval between the Event Time Point (ETP) and the following Update Time Point (UTP). The host has moved but the participants are not notified. Consequently, all sent packets in DI get lost, since the sender uses malicious IP or port.

The *Maximal Disconnection Interval* (MDI) is the maximal disconnection tolerable by the application. For real time applications, the MDI can be some milliseconds. The MDI can be several minutes for services, like e-mail. The DI must not exceed the MDI.

A. Time points relative to filter cycle

The filters work with sequential execution of prediction and update phase, which build one filter cycle. In the prediction phase, prior estimation of the PDF of EIs is made using the past disconnections. The UTPs are calculated using this PDF. In the update phase, the updates are executed until the result of the update becomes *false*, i.e. UI with ETP. The posterior estimation is made using the result. Then the filter cycle starts again. All time points are set relative to the beginning of the filter cycle in order to avoid working with absolute values. The filter cycle begins with first UTP, thus zero point. All following ETPs and UTPs in the cycle are set relative to it.

VI. Optimization Task

The Disconnection Interval (DI) must be minimized. Unfortunately, the Event Time Point (ETP) is unknown to the mobile host and therefore, the DI cannot be calculated. The idea is to minimize the UI with ETP, which is known to the mobile application. The DI is smaller than UI with ETP, thus decreasing the UI with ETP minimizes the DI.

To save recourses, the UI without ETP must be maximized. The DI must not exceed the Maximal Disconnection Intervals (MDI) defined by the application. Consequently, the UI must also not exceed the MDI. The UI without ETP must be equal (best case) or less than MDI.

VII. Model

Every filter requires the definition of two models. The first one describes the measurement of the variable of interest, like measurement of the velocity. The second one describes the evolution of the variable of interest (natural properties), velocity of the falling body.

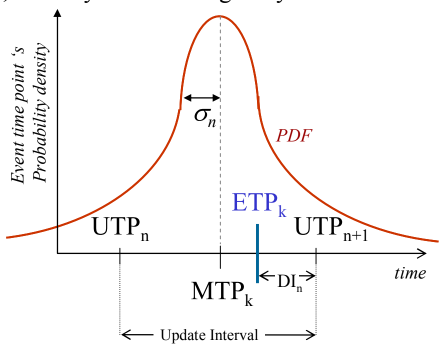

Figure 3 Measurement model

Some assumptions are made in order to define these two models. First, the Measured Time Point (MTP) of ETP is assumed to be in the middle of the UI, so that the absolute error is minimal. Second, the probability for Event Time Point in the UI has Normal distribution. The mean of the Normal distribution is at MTP. The standard deviation is set proportional to the size of the UI by *stretching* factor (constant). The distribution is zero outside the UI interval. The Figure 3 shows the measurement model.

The measurement is expressed using *f()* function. The function returns the MTP, thus middle of the Update Interval with ETP. A second returned argument of the function is the maximal measurement error, thus the UI with ETP. The model equation is:

$$[y_k, e_k] = f(x_k)$$

$$p(y_k|x_k) = N[y_k, c_{st} \cdot e_k],$$

where y_k is the measured ETP (estimated) and e_k denotes the maximal error (UI size). The x_k is the ETP. The second equation expresses the Probability Distribution Function. The $N[\ ]$ is the Normal distribution with mean y_k and standard deviation $c_{st} \cdot e_k$. The stretching coefficient is c_{st}.

The evolution of ETP cannot be defined, since the movements of the host are unknown. Any defined model restricts the practical scenarios. For this reason, it is assumed zero-knowledge. The prior estimation is the same as the posterior estimation, thus prediction is equal to the measurement. This assumption leads to degeneracy of particle positions, which is avoided through particle moving (see section X).

VIII. Particle filer for Updates optimization

The description on particle filter, also known as Sequential Monte Carlo, can be found in [3]. The focus in this section is the derivation of the new methods.

The PDF built by the Update Time Points Ru_i for M points, is:

$$p(u)_M = \frac{1}{M}\sum_{i=1}^{M}\delta(u - Ru_i),$$

where the subscript denotes the filter cycle.

Let us denote the PDF of EI as *p(t)*. In the particle filter, the PDF is presented by *N* particles. Each particle consists of two values: position v and weight w. The PDF is then $\{v^i, w^i\}_{i=1}^{N}$ The weights stress the high values of the PDF and are normalised, thus $\sum w^i = 1$. The PDF of EI can be expressed as:

$$p(t) \xrightarrow[N\to\infty]{} \frac{1}{N}\sum_{i=1}^{N} w^i\delta(t - v^i)$$

The PDF of EI defined by the particle filter, is:

$$p(t)_N = \frac{1}{N}\sum_{i=1}^{N} w^i\delta(t - v^i),$$

where the v^i is the position of the particle and *N* the total number of the particles.

The PDF by EI should be proportional to the PDF by the UTP within the filter cycle as defined in II. This is an ideal case, which can be expressed with:

$$p(t)_N = \frac{1}{N}\sum_{i=1}^{N} w^i\delta(t - v^i) \propto p(u)_M = \frac{1}{M}\sum_{i=1}^{M}\delta(u - Ru_i)$$

This is a key equation for calculating the UTP. Notice that the PDF functions must be only proportional.

Observing the last equation, the closest suggestion is to set the Update Time Points (UTP) equal to the particle values (positions). Where the concentration of particle is high, there will be a concentration of updates respectively. It is absolutely correct and reasonable from a mathematical perspective. Unfortunately, the Update Intervals must not exceed the Maximum Disconnection Interval (MDI) defined by the user (section VI). A direct copy of the values can not satisfy this condition.

The condition UI < MDI gives that the PDF of UTPs could not be exact proportional to the PDF by EIs, since there must be updates also in arias with zero probability of ETP. Following, the condition of PDF proportionality is ideal one and not achievable. The practical condition is that the PDFs must be proportional in some degree.

A practical suboptimal approach is suggested for achieving this. It is easy to implement and delivers excellent result in the simulation (section XI).

At first step, an Update Time Point(UTP) is set at every $x_{bin} \leq x_{\max}$, where $x_{\max}$ is the Maximum Disconnection Interval. The condition for maximum disconnection is fulfilled in this way. The resulting intervals are called bins. At second step, a number of uniform distributed UTP is added in each interval x_{bin} (bin) depending on the PDF built by the particle. If the PDF is high in the bin, then a large number of UTP points is added. Certainly, the PDF is not a constant in the bin intervals. The reference the value is in the middle of the bin. Different reference values can be considered involving interpolation, mean etc. This is out of scope in this general method description.

The total number of additional UTPs added in all bins, l_{LUsum}, is defined by the user before the simulation begins. The higher the number added UTPs the lower the disconnection time and the lower the measurement error. Unfortunately, the higher the number of added updates the more resources are required. The performance of the algorithm becomes poor because of the wasted resources. If there are zero added UTPs, then there are constant intervals. as if no PDF considerations are done. In current experience, values between 20 and 200 added updates for 10 seconds of MDI deliver good results.

Number of added UTPs in the bin is calculated by multiplying the PDF value at the middle of the bin by the total number of added UTPs. The PDF must be normalised before calculation (sum of all particle weights must be one). The PDF value in the middle of the bins is denoted as $\{h_i\}_1^{bins}$. The number l_i of added updates in the ith bin will be:

$$l_i = h_i l_{LUsum}$$

The distribution of the additional updates in the bin is uniform. There is no other information about the distribution, so uniform distribution seems to be the proper one.

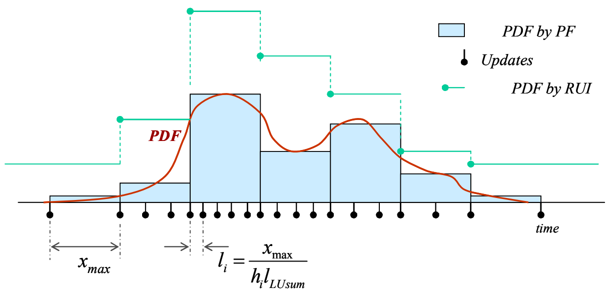

Figure 4 Updates distribution

The updates and the PDF function are shown at Figure 4. The PDF in bin is approximated to constant values equal to the middle PDF values in the bin. The uniform distribution of the updates dependents of bins high (PDF value in the middle). The size of the update intervals in the bin is division of x_{bin} to the number of added UTPs in bin:

$$u_i = \frac{x_{bin}}{h_i l_{LUsum}}$$

IX. Particle Update

The main goal of the particle update procedure is to evaluate new weights using the measurement, posterior estimation of the PDF. The weights represent the PDF function and the positions are concentrated at the high values of the PDF curve. The particle update is multiplication of the previous weights with the conditional probability [2, 3]:

$$w_k^{*i} = w_{k-1}^{*i} p(y_k | x_k^i)$$

Using multiplication has the advantage that normalization do not influence the relation between the coefficient. The factorization (multiplication) is suitable for sequential update of arguments. A disadvantage of the multiplication with a Gaussian importance function is the extreme decrease of the coefficients at the low Gaussian values. The variance of multiplication increases with every interaction. This is the main reason for the degeneracy of the weights in particle filter [6].

The PDF of EI must be proportional to the PDF of the UTPs as already mentioned. The exact equality is out of scope. Making use of this fact, the multiplication can be exchanged by addition, thus:

$$w_k^{*i} = w_{k-1}^{*i} + p(y_k | x_k^i)$$

The big advantage is that there is no degeneracy of the weight coefficients. The variance does not increase rapidly. The resulting function is proportional to the PDF and there is no degeneracy. A major disadvantage of the addition is the permanent commutation of the coefficient values. The values can only increase. Modern computers work with finite numbers, so the hardware will be overloaded at some point. Normalisation is a way of solving the increasing in the weights.

$$w^i = \frac{w^{*i}}{\sum w^{*i}}$$

Unfortunately, the normalisation in this case changes ratio between the updated coefficients. The prior updates become less important to all posterior following updates after normalisation. The weights are not treated fair. This follows straightforward from the mathematical equation for normalisation. For example: one weight after two normalisations at step *k-1* and *k-2* is:

$$w_k^i = \frac{w_{k-1}^i + p(y_{k-1} | x_{k-1}^i)}{\sum_i w_{k-1}^i} = \frac{\frac{w_{k-2}^i + p(y_{k-2} | x_{k-2}^i)}{\sum_i w_{k-2}^i} + p(y_{k-1} | x_{k-1}^i)}{\sum_i w_{k-1}^i}$$

$$= \frac{w_{k-2}^i + p(y_{k-2} | x_{k-2}^i)}{\sum_i w_{k-2}^i \sum_i w_{k-1}^i} + \frac{p(y_{k-1} | x_{k-1}^i)}{\sum_i w_{k-1}^i}$$

The terms $p(y_{k-2} | x_{k-2}^i)$ and $p(y_{k-1} | x_{k-1}^i)$ are divided to different divisors, thus the updates are not considered fair.

There must be as little normalisation as possible to avoid unfair treatment. The normalisation is carried out only when the coefficients reach the threshold to overload.

Using addition instead of multiplication increases the performance of the algorithm. There is no degeneration of the weights or the particles. The solution is acceptable, since the UTP distribution procedure only requires proportionality to the real PDF by. The effect of normalisation does not lead to underperformance, as the simulation shows.

An interesting fact is that normalisation on regular basis can be considered a forgetting factor. This could have positive influence in an environment with dynamic PDF.

X. Particle Moving

The particle positions (values) are very important for building the PDF. Please notice that the particles must not be distributed as the PDF function since the information for the function is carried by the weights. The positions define which representative points are chosen to approximate the function. There are two requirements for the particle positions. First, the number of particles must be sufficient to represent the PDF. Second, the particles must be distributed at the higher points of the PDF. The distribution of the particles is very important. For example: if the PDF has almost zero up to 1000 sec, that there is no sense of keeping positions at less than 1000 sec.

With zero knowledge and no use of resampling, the particle positions are constant during the simulation. To overcome this shortcoming, an algorithm has been developed for moving of particle. Moving is a synonym for changing the position value of particle.

The principle is that if the there is not particle value closer than x_{bin} to the measured value (MTP) than the particle with the lowest weight is set to measured value. The weight of the moved particle is set to zero. The procedure is executed before updating the weights.

The algorithm moves the particle with the lowest weight to cover the PDF high values. The high values of the PDF are represented by large weights. Only particles with lower weights are moved. The distance size of x_{bin} is considered sufficient, since the UTPs are uniformly distributed (constant) in the bin of x_{bin}.

XI. Simulation

The main target of the simulation is to achieve proof of concept and to show the qualities of the new method. The performance of the new method is compared to the constant interval update in a fair way. The fair way means the same number of updates for the same simulation. The two methods use the same total number of updates, input values and simulation time. The better method achieves less disconnection, thus better estimation. The constant updates is commonly the only used method, so it is compared the current protocols to the new one.

The constant parameters in the simulation are chosen to represent real cases, where small disconnection is required and the mobile host moves fast. For the experiments, the following parameters are used: The application requires 5 sec of Maximum Disconnection Interval (MDI). The simulation is made with 5000 ETPs (filter cycles). There are 100 particles for constructing the PDF. The added updates are 100 and the stretching coefficient is 3.

The simulation shows the clearly the outperformance of the new method. Here some representative cases are shown.

A. *Non linear EI with white noise*

The Event Intervals are generated with recursion, where the next value depends of the previous one:

$$aEI_k = \frac{aEI_{k-1}}{2} + \frac{25r_{k-1}}{1+r_{k-1}^2} + 8\cos(1.2k) + N(200{,}20)$$

The EI and the estimated EI are shown at Figure 5. The results in at Figure 6 show a very good performance of the algorithm. The abbreviations are:

- A dotted black line shows the mean Update Interval (UI) by the new method, thus mean of maximal disconnection.
- A dashed black line shows the maximal Disconnection Intervals (DIs) by constant update interval, see variable *const UI*.
- A red dashed line marks the maximum Disconnection Interval (error) by the new method.
- A grey dotted line shows the user defined Maximum Disconnection Interval (MDI).

Certain very important qualitative values are shown in the bottom-right corner of the histogram (Figure 6). Their definitions are:

- {mean DI} is mean of all Disconnection Intervals.
- {mean UI} is the mean of the Update Intervals containing the Event Time Point.
- {max UI} is the maximum of all Update Intervals during the simulation.
- {max user def DI } is the Maximum Disconnection Time.
- {const UI}. This is the maximum disconnection achieved by constant update method using the same resources as the new method.
- {const UI mean DI } is the mean of DI by constant UI with the same resources.
- {const UI / mean UI} gives the ratio between the *const UI* and *mean UI*. This is a direct comparison of the performance of the algorithm in a simple way.
- {LU < const UI}. is the number in percentage of UI with ETP smaller than the maximum disconnection by constant interval (*const UI*).

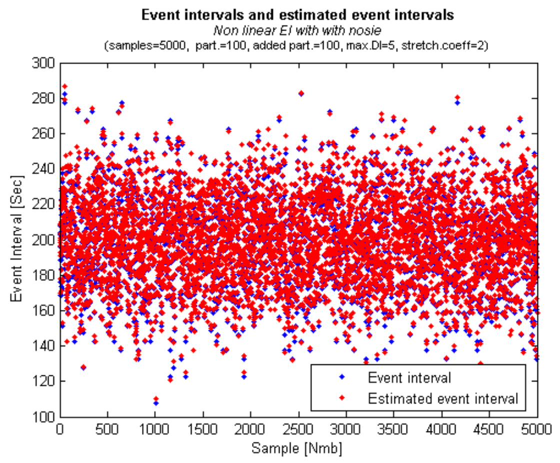

Figure 5 Non-linear data, EIs and estimated EIs

The mean UI by the new method is 2.19 times smaller than the mean UI by the constant interval. The 91,2% percents of the prediction cycles deliver better result than the constant updates. The MDI is reached in small number of cases as shown in Figure 6. There is clear outperformance of the new method over the constant upates.

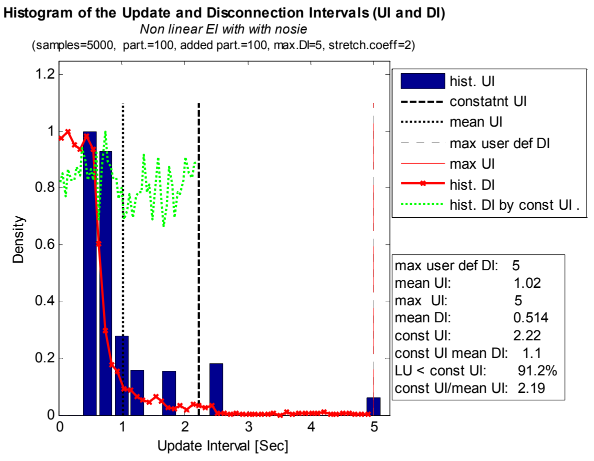

Figure 6 Non-linear data, histogram of the results

B. *Sinus based EI with white noise*

For this simulation is a sinus based signal used with white noise:

$$aEI_k = \sigma \sin\left(4\pi.\frac{k}{S}\right) + \frac{7}{10}\sigma \sin\left(\frac{3}{2}\pi\frac{k}{S}\right) + \\ + \sin c\left(\frac{k}{S}\right) + N\left(500, \frac{\sigma}{2}\right)$$

The variable S indicates the total number of samples, S=5000. The *k* is the index of the calculated sample. At Figure 7 the EI are presented.

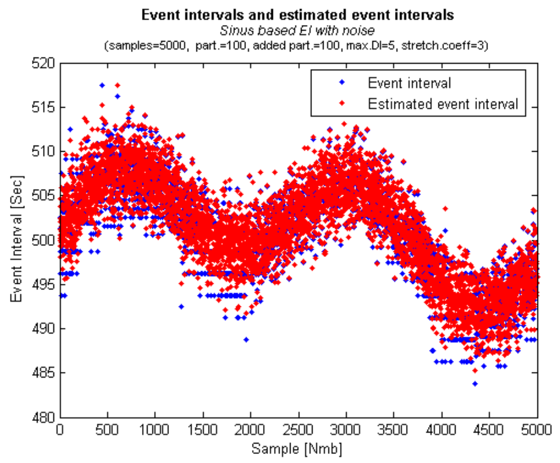

Figure 7 Sinus based data, EI and estimated EIs

The Figure 10 presents the result histogram. The results show that this is an even better case for the new algorithm. The 99.8% of the updates are better then the constant update interval. The constant UI is 3.52 times bigger then the mean UI. This is generally due to the bigger values of Event Interval, compared to the previous simulations, the values lay around 500 sec

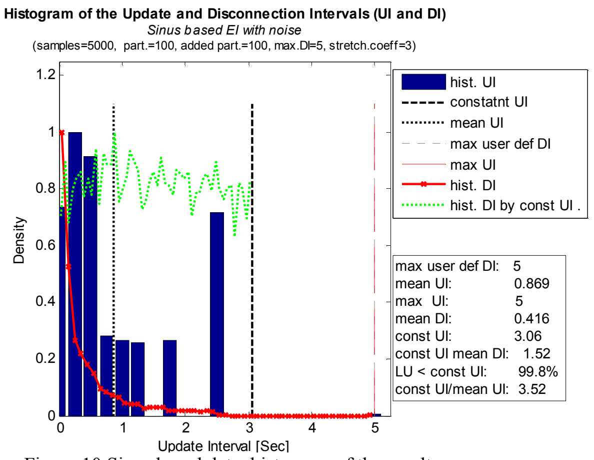

Figure 10 Sinus based data, histogram of the results

C. *Two rotating white-noise sources*

There are two white-noise sources, which are rotating. They can be described by:

$$aEI_r = N(100,\sigma_1)\ or\ N(100,\sigma_2)$$
$$\sigma_1 = 10,\ \sigma_2 = 10$$

The operator *or* denotes the rotation with 50% probability. Half of the values are generated by the first distribution and half by the second. The results are shown at Figure 9.

The mean UI is 2.24 smaller then the constant UI. 93.7% of the updates are smaller than the UI by constant updated. The PDF constructed by the particle filter and the PDF by EI are shown at Figure 8, where the PDF are scaled by factor to achieve equal maximum value. It can be concluded that the new method performs independent of the PDF form, as designed.

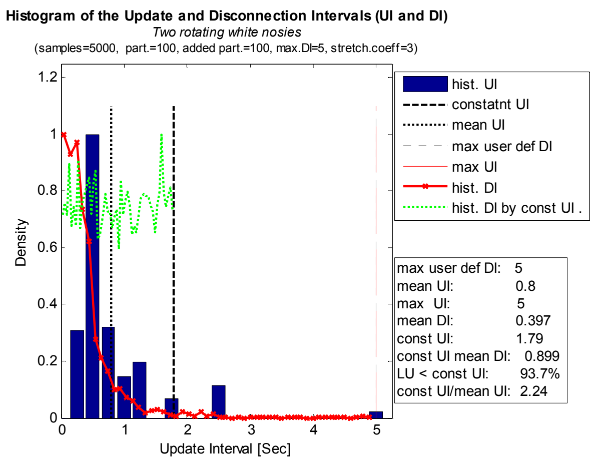

Figure 9 Two white-noise sources, histogram of the results

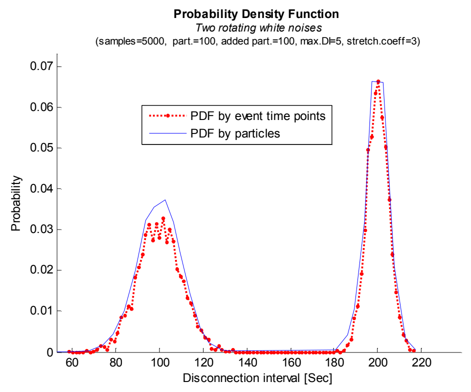

Figure 8 Two white-noises sources, PDF by EI and by particle

XII. Conclusion

The new method has significant outperformance over the current constant update intervals method. The method profits history of the past event to adjust the update frequency. It can be deployed multiple protocols which are used in mobile environment. The method is an alternative of Fuzzy controller suggested in [8].

References

[1] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear and non-Gaussian Bayesian state estimation", Proc. Inst. Elect. Eng., F, vol. 140, pp. 107–113, 1993.

[2] M. S.Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking", IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 50, NO. 2, 2002

[3] A. Doucet, SJ Godsill and C. Andrieu, "On Sequential Monte Carlo sampling methods for Bayesian filtering", Stat. Comp., 2000

[4] Keith Copsey, "Tutorial on Particle filters", Pattern and Information Processing Group,DERA Malvern, Jan 2001

[5] N.J. Gordon, L. Louise, "Presentation General PF discussion" , October 2003

[6] Paul Fearnhead, Merton College, "Sequential Monte Carlo methods in filter theory", University of Oxford, PhD Thesis, 1998

[7] Niclas Bergman, Linkoping, "Recursive Bayesian Estimation Navigation and Tracking Applications", Studies in Science and Technology. Dissertations, No. 579, 1999

[8] V. Tzvetkov, "Optimization of update intervals in Dead-Peer-Detection using adaptive Fuzzy Logic", IEEE AINA, 2007