Distributed Target Tracking With Optimal Data Migration.

Vahram Stepanyan *
NASA Ames Research Center, KBR Wyle Services, Moffett Field, CA 94035

Keerthana Kannan†
NASA Ames Research Center, KBR Wyle Services, Moffett Field, CA 94035

Evan T. Kawamura‡
NASA Ames Research Center, Moffett Field, CA 94035

Thomas Lombaerts§
NASA Ames Research Center, KBR Wyle Services, Moffett Field, CA 94035

Corey Ippolito ¶
NASA Ames Research Center, Moffett Field, CA 94035

The paper presents an Extended Kalman Filter based framework for airborne target tracking using dynamic information fusion from multi-modal sensors with geodiversity. First, the algorithm execution location is determined using an optimal data migration strategy, next the sensors information is dynamically fused at each estimation instance using validity flag for each sensor reading, finally the target estimation is updated based on the fused innovation vector. The approach is applied to synthetic data generated from the radar and camera models located on the ground for the simulated target flight in Reflection simulation environment.

I. Problem Formulation

We consider a target tracking problem using information from distributed multi-modal sensors $s_i,\ i = 1,\ldots,N_s$, which can be both static and mobile ground based or airborne. We assume that the sensors positions $\boldsymbol{p}_i^s\ i = 1,\ldots,N_s$ in a Cartesian ENU (East-North-Up) frame F_0 and the rotation matrices $R_i,\ i = 1,\ldots,N_s$ from sensor-attached frames $F_i,\ i = 1,\ldots,N_s$ to F_0 are known. Sensors transmit the target's possibly preprocessed information over the wired or wireless network (assumed to be cellular) as soon as it becomes available. This preprocessing may include data classification, compression, encryption or complete processing depending on the type of sensor. For example, in the case of radar, this may include obtaining target's range, azimuth and elevation information from the sensor readings. In any case, we assume that the size of data to be transmitted is known.

Data is received and processed at computing centers $c_i,\ i = 1,\ldots,N_c$ using information fusion and target tracking algorithms. It is assumed that the locations $\boldsymbol{p}_i^c\ i = 1,\ldots,N_c$ in the frame F_0 and computing capacities of these centers are known. The computing centers are connected via wired or wireless network, depending on the available infrastructure and the regional data processing demand. Since data sources have diverse locations, the information fusion and target tracking algorithms have to deal with the latency in communications and processing

The objective is to design algorithms for data migration to a computing center that provides minimum end-to-end latency, for fusion of information from all sensors as it becomes available, and for tracking the target based on dynamically resizing observations.

II. Optimal Data Migration

We estimate the elapsed time based on the available best estimates of the sensors locations, data pre-processing capabilities at sensors stations, communication networks transmission rates and bandwidths, and algorithms execution capabilities at computing centers.

*Technical Advisor, KBR Wyle Services, Intelligent Systems Division, Mail Stop 269-1, email: vahram.stepanyan@nasa.gov
†Aerospace Research Engineer, KBR Wyle Services, Intelligent Systems Division, Mail Stop 269-1, email: keerthana.kannan@nasa.gov
‡Aerospace Research Engineer, Intelligent Systems Division, email: evan.t.kawamura@nasa.gov
§Aerospace Research Engineer, KBR Wyle Services, Intelligent Systems Division, Mail Stop 269-1, email: thomas.lombaerts@nasa.gov
¶Aerospace Research Engineer, Intelligent Systems Division, Mail Stop 269-1, email: corey.a.ippolito@nasa.gov

Let the data rates (in bits/second) for wired communication between the stationary infrastructure units (sensor stations, communication towers and computing centers) be ω, which is assumed to be constant for all wired communications legs. Let the input data of size $D(\cdot)$ bits be transmitted for a distance $d(s_{\cdot_i}, s_{\cdot_j})$ between stationary units $s_{\cdot_i}$ and $s_{\cdot_j}$. According the model introduced in Ref. [1], the average communication latency of this data transmission is given by

$$\tau_w(D(\cdot), \cdot_i, \cdot_j) \quad = \quad \frac{d(s_{\cdot_i}, s_{\cdot_j})}{c} + \frac{D(\cdot)}{\omega}, \tag{1}$$

where c is the propagation speed (in meter/second) of electromagnetic waves. We assume that the communication tower T_i processes data as they arrive trough a wireless channel with the rate of λ_T in terms of CPU cycles per second. Then the processing latency is computed as

$$\tau_{pr}(D(\cdot), T_i) \quad = \quad \frac{D(\cdot)}{\alpha_T \lambda_T}, \tag{2}$$

where α_T stands for the number of bits processed per one CPU cycle.

We assume a non-orthogonal multiple access (NOMA) protocol for the wireless data transmission between mobile sensors m_{s_i} or mobile computation centers m_{c_i} and communication towers T_k, which allows all users to simultaneously utilize the entire system bandwidth with a multiple access interference in the coverage area [3, 4], but neglects the interference between adjacent cells. We denote $m_{s_i} \in H(T_k)$ and $m_{c_i} \in H(T_n)$ to indicate that the mobile sensor m_{s_i} and mobile computing centers m_{c_i} are in the coverage cell of towers T_k and T_n respectively. Then the up-link transmission rates between m_{s_i} and T_j and down-link transmission rate between T_k and m_{c_i} are respectively expressed as

$$\begin{aligned}
\rho_u(m_{s_i}, T_j) \quad &= \quad B_{sys}^j \log_2 \left(1 + \frac{p(m_{s_i}, T_j) g m_{s_i}, T_j)}{\sigma_r^2 + \sum_{h=1,\ h \neq i, m_{\cdot_h} \in H(T_j)}^{h=N_m} p(m_{\cdot_h}, T_j) g(m_{\cdot_h}, T_j)} \right) \\
\rho_d(T_k, m_{c_i}) \quad &= \quad B_{sys}^k \log_2 \left(1 + \frac{p(T_k, m_{c_i}) g(T_k, m_{c_i})}{\sigma_r^2 + \sum_{h=1,\ h \neq i, m_{\cdot_h} \in H(T_j)}^{h=N_m} p(T_k, m_{\cdot_h}) g(T_k, m_{\cdot_h})} \right),
\end{aligned} \tag{3}$$

where N_m is number of mobile units, B_{sys}^j is the network bandwidth associated with the tower T_j, $p(m_{\cdot_h}, T_j)$ and $p(T_j, m_{\cdot_h})$ are respectively the transmitting powers from mobile unit $m_{\cdot_h}$ to tower T_j and vice-versa, $g(m_{\cdot_h}, T_j)$ and $g(T_j, m_{\cdot_h})$ are respectively the corresponding transmission channel power gains, and σ_r^2 is the power of additive white Gaussian noise at the receiver. The channel power gains $g(m_{\cdot_h}, T_j)$ and $g(T_j, m_{\cdot_h})$ depend on the transmitter to receiver Euclidean distances $d(m_{\cdot_h}, T_j)$ as

$$g(m_{\cdot_h}, T_j) = g(T_j, m_{\cdot_h}) = \frac{g_0}{d^2(m_{\cdot_h}, T_j)}, \tag{4}$$

where g_0 is the reference gain [2]. Then corresponding up-link and down-link data transmission latency is computed for each data volume $D(\cdot)$ as

$$\tau_u(D(\cdot), m_{\cdot_h}, T_j) = \frac{D(\cdot)}{\rho_u(m_{\cdot_h}, T_j)}, \quad \tau_d(D(\cdot), T_j, m_{\cdot_h}) = \frac{D(\cdot)}{\rho_d(T_j, m_{\cdot_h})}. \tag{5}$$

For the execution time estimation, we assume that the computing center c_i has a data processing capability of λ_{ci} cycles per second and α_{ci} bits per one CPU cycle. Then the center c_i will spend

$$\tau_{pr}(D(\cdot), c_i) = \frac{D(\cdot)}{\alpha_{ci} \lambda_{ci}} \tag{6}$$

seconds to process a data of size $D(\cdot)$ and

$$\tau_e(a, c_i) = \frac{A}{\lambda_{ci}} \tag{7}$$

second to execute the estimation algorithm, which is assumed to have a size of A in terms of CPU cycles. To estimate the total elapsed time, we separately compute the time for each input data to reach from all sensors to the execution location, take the maximum of them and add the service execution time. That is for a mobile computing center $m_{c_n} \in H(T_k)$ we obtain

$$
\begin{aligned}
\tau(m_{s_i}, m_{c_n}) &= \tau_u(D(m_{s_i}), m_{s_i}, T_j) + \tau_{pr}(D(m_{s_i}), T_j) + \tau_w(D(m_{s_i}), T_j, T_k) && (8)\\
&+ \tau_{pr}(D(m_{s_i}), T_k) + \tau_d(D(m_{s_i}), T_K, m_{c_n}) && (9)\\
\tau(s_{s_j}, m_{c_n}) &= \tau_w(D(s_{s_j}), s_{s_j}, T_k) + \tau_{pr}(D(s_{s_j}), T_k) + \tau_d(D(s_{s_j}), T_K, m_{c_n}) && (10)\\
\tau(m_{c_i}, m_{c_n}) &= \tau_u(D(m_{c_i}), m_{c_i}, T_j) + \tau_{pr}(D(m_{c_i}), T_j) + \tau_w(D(m_{c_i}), T_j, T_k) && (11)\\
&+ \tau_{pr}(D(m_{c_i}), T_k) + \tau_d(D(m_{c_i}), T_K, m_{c_n}) && (12)\\
\tau(s_{c_j}, m_{c_n}) &= \tau_w(D(s_{c_j}), s_{c_j}, T_k) + \tau_{pr}(D(s_{c_j}), T_k) + \tau_d(D(s_{c_j}), T_K, m_{c_n}) && (13)\\
\tau_{tr}(m_{\cdot_i}, m_{c_n}) &= \max\{\tau(m_{s_i}, m_{c_n}), \tau(m_{c_i}, m_{c_n}) : i = 1, \ldots, N_m\} && (14)\\
\tau_{tr}(s_{\cdot_i}, m_{c_n}) &= \max\{\tau(s_{s_i}, m_{c_n}), \tau(s_{c_i}, m_{c_n}) : i = 1, \ldots, N_s\} && (15)\\
\tau_{tr}(m_{c_n}) &= max\{\tau_{tr}(m_{\cdot_i}, m_{c_n}), \tau_{tr}(s_{\cdot_i}, m_{c_n})\} && (16)\\
\tau(m_{c_n}) &= \tau_{tr}(m_{c_n}) + \tau_e(a, m_{c_n}), && (17)
\end{aligned}
$$

where N_s is number of stationary units, and for the stationary computing center s_{c_n} we obtain

$$
\begin{aligned}
\tau(m_{s_i}, s_{c_n}) &= \tau_u(D(m_{s_i},), m_i, T_j) + \tau_{pr}(D(m_{s_i},), T_j) + \tau_w(D(m_{s_i},), T_j, s_{c_n}) && (18)\\
\tau(s_{s_j}, s_{c_n}) &= \tau_w(D(s_{s_j}), s_{s_j}, s_{c_n}) && (19)\\
\tau(m_{c_i}, s_{c_n}) &= \tau_u(D(m_{c_i}), m_{c_i}, T_j) + \tau_{pr}(D(m_{c_i}), T_j) + \tau_w(D(m_{c_i}), T_j, s_{c_n}) && (20)\\
\tau(s_{c_j}, s_{c_n}) &= \tau_w(D(s_{c_j}), s_{c_j}, s_{c_n}) && (21)\\
\tau_{tr}(m_{\cdot_i}, s_{c_n}) &= \max\{\tau(m_{s_i}, s_{c_n}), \tau(m_{c_i}, s_{c_n}) : i = 1, \ldots, N_m\} && (22)\\
\tau_{tr}(s_{\cdot_i}, s_{c_n}) &= \max\{\tau(s_{s_i}, s_{c_n}), \tau(s_{c_i}, s_{c_n}) : i = 1, \ldots, N_s\} && (23)\\
\tau_{tr}(s_{c_n}) &= max\{\tau_{tr}(m_{\cdot_i}, s_{c_n}), \tau_{tr}(s_{\cdot_i}, s_{c_n})\} && (24)\\
\tau(s_{c_n}) &= \tau_{tr}(s_{c_n}) + \tau_e(a, s_{c_n}), && (25)
\end{aligned}
$$

We notice that data compression, encoding and decoding or other type of processing can be computed as in (2).

Now we formulate an optimization problem to determine the data and estimation algorithm migration policy, which includes an initial allocation as well as consequent migration as the sensors or computing centers move. Our goal is to design a decision making algorithm to send the data and estimation algorithm itself to the computing center while minimizing the latency $\tau(c_i), i = 1, \ldots, N_c$ as estimated in (8) and (18). To this end, we introduce binary variables $x_i, i = 1, \ldots, N_c$, which take on value $x_i = 1$ when the target tracking is placed in the computing center c_i for execution and zero otherwise. To make sure that only on location can be selected, we impose a constraint

$$\sum_{i=1}^{N_c} x_i = 1, \tag{26}$$

on the binary variables x_i. The cost function to be minimized is expressed as

$$f = \sum_{i=1}^{N_c} x_i \tau(c_i). \tag{27}$$

The user imposed latency upper limit, if any, is enforced via equation

$$\sum_{i=1}^{iN_c} x_i \tau_i \le \tau_{\max}. \tag{28}$$

The optimization problem is a mixed-integer linear program

$$
\begin{aligned}
P1: \quad & \min\ f && (29)\\
& s.t. \quad 0 \le x_i \le 1,\ i = 1, ..., N_c,\ and\ (26),\ (28).
\end{aligned}
$$

with possible additional constraints on available computational, storage and power resources, which are not detailed here.

III. Extended Kalman Filter Framework

We use a conventional continuous time coordinate-uncoupled white-noise acceleration model for a point object to describe the target's motion is in Cartesian frame F_0 [5], which is disctretized with a time step δt.

$$\boldsymbol{s}(k) = \Phi(k-1)\boldsymbol{s}(k-1) + \Gamma(k-1)\boldsymbol{w}(k-1)\,, \tag{30}$$

where the vector $\boldsymbol{s}(k) = [x(k)\ \dot{x}(k)\ y(k)\ \dot{y}(k)\ z(k)\ \dot{z}(k)]^\top$ represents the target states $\boldsymbol{w}(k) = [n_x(k)\ n_y(k)\ n_z(k)]^\top$ is the input noise vector, and

$$\Phi(k) = \begin{bmatrix} 1 & \delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \ \Gamma(k) = \begin{bmatrix} \delta t^2/2 & 0 & 0 \\ \delta t & 0 & 0 \\ 0 & \delta t^2/2 & 0 \\ 0 & \delta t & 0 \\ 0 & 0 & \delta t^2/2 \\ 0 & 0 & \delta t \end{bmatrix}$$

are the state transition and input matrices.

The observation models are sensor specific and have variable sizes. In the case sensor s_i is a radar, the observation model is given in $3D$ spherical coordinate system $\{r,\ \theta,\ \phi\}$ as

$$\begin{aligned} r_i^m &= r_i + b_i r_i + n_{ri} \\ \theta_i^m &= \theta_i + n_{\theta i} \\ \phi_i^m &= \phi_i + n_{\phi i}\,, \end{aligned} \tag{31}$$

where superscript m indicates the measured values of the corresponding quantities, $n_{ri},\ n_{\theta i}\ n_{\phi i}$ are assumed to be zero-mean, Gaussian distributed, and uncorrelated processes with variances $\sigma_{ri}^2,\ \sigma_{\theta i}^2,\ \sigma_{\phi i}^2$, and b_i is assumed to be an unknown constant. This measurements relate to the target's coordinates in the sensor frame F_i as

$$\begin{aligned} r_i &= \sqrt{x_i^2 + y_i^2 + z_i^2} \\ \theta_i &= tan^{-1}\left(\frac{y_i}{x_i}\right) \\ \phi_i &= tan^{-1}\left(\frac{z_i}{r_{ih}}\right), \quad r_{ih} = \sqrt{x_i^2 + y_i^2}\,, \end{aligned} \tag{32}$$

and $[x_i\ y_i\ z_i]^\top = R_i^\top\left([x\ y\ z]^\top - \boldsymbol{p}_i^s\right)$. In the case sensor s_i is a camera, the observation model is given in $2D$ image coordinate system $\{x_p,\ y_p\}$ as pixel measurements

$$\begin{aligned} x_{pi}^m &= x_{pi} + n_{pxi} \\ y_{pi}^m &= y_{pi} + n_{pyi}\,, \end{aligned} \tag{33}$$

where superscript m indicates the measured values of the corresponding quantities, $n_{pxi},\ n_{pyi}$ are assumed to be zero-mean, Gaussian distributed, and uncorrelated processes with variances $\sigma_{pxi}^2,\ \sigma_{pyi}^2$. This pixel coordinates relate to the target's position in the sensor frame F_i as

$$\begin{aligned} x_{pi} &= -\frac{x_i}{z_i}\frac{f_i}{dp_i} + \frac{W_i}{2} \\ y_{pi} &= -\frac{y_i}{z_i}\frac{f_i}{dp_i} + \frac{H_i}{2}\,, \end{aligned} \tag{34}$$

where f_i, dp_i, W_i, H_i are the camera focal length, pixel size, camera width and camera height in pixels respectively, and $[x_i\ y_i\ z_i]^\top = R_i^\top\left([x\ y\ z]^\top - \boldsymbol{p}_i^s\right)$. Depending on the sensors readings validity we make an observation vector $\boldsymbol{h}(x,y,z)$ stacking up the measurements and compute the corresponding Jacobean matrix $H(\hat{x},\hat{y},\hat{z})$ to be used in Kalman gain computation at each estimation step, where $\hat{x},\ \hat{y},\ \hat{z}$ are the estimates of the target coordinates in F_0. This estimation process with the use of Extended Kalman Filter (EKF) including the estimation of unknown constants b_i will be detailed in the final paper.

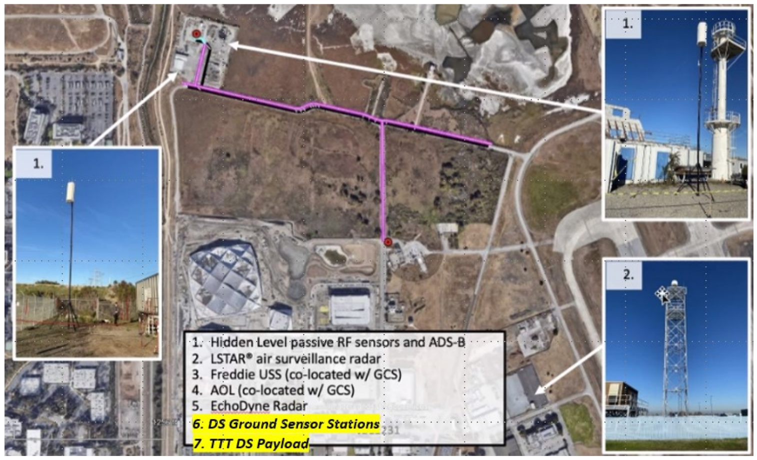


Fig. 1 Target's trajectory on the flight test site map

IV. Preliminary Simulation Results

For this preliminary simulation we use the UAV planned trajectory for the test flight at NASA Ames Research Center's DART facility as the target's motion, which is simulated in Reflection simulation environment. This trajectory is displayed in Fig. 1. It is also used to generate sensors readings using in-house developed radar and camera models with selected parameters and noise characteristics (will be detailed in the final version). We allocate three radars and four cameras in the DART site, the locations of which are fixed for this preliminary simulation study and are displayed in Fig 2 as yellow circles, where the computing centers are also displayed as green circles and assumed to be fixed. The coordinates of these locations are obtained via Google Earth.

The generated radar and camera readings along with the validity flag that indicates the target is in the sensor's field of view are used to track the target using EKF framework with dynamically resizing measurement vector as presented in Section III. Fig. 3 displays the tracking errors for the first second of flight, where good convergence properties of the proposed approach can be observed.

Fig. 4 displays the tracking errors for the cruising flight, where the bursting can be attributed to targets turning maneuvers.

Fig. 5 displays the tracking errors for the last 8 *sec* of flight, when the target is landing almost vertically. The large errors observed in x and y direction can be attributed to lack of the useful information in the sensors readings in x and y direction.

The final version will include more detailed simulation description and analysis, including sensors and computing centers mobility and possibly the flight test results for the real time target tracking.

Acknowledgment

This work was performed under the NASA Aeronautics Research Mission Directorate (ARMD), Transformative Tools and Technologies (TTT) Project's Autonomous Systems activity.

References

[1] K. Miller. Communication Theories: Perspectives, Processes, and Contexts. *New York, NY, USA: McGraw-Hill, 2005*.

[2] S. Jeong, O. Simeone, and J. Kang. Mobile edge computing via a UAV mounted cloudlet: Optimization of bit allocation and path planning. *IEEE Trans. Veh. Technol., vol. 67, no. 3, pp. 2049–2063, Mar. 2018*.

[3] L. Dai, B. Wang, Y. Yuan, S. Han, C. -L. I, and Z. Wang. Non-orthogonal multiple access for 5G: Solutions, challenges, opportunities, and future research trends. *IEEE Commun. Mag., vol. 53, no. 9, pp. 74–81, Sep. 2015*.

[4] S. Hu and G. Li. Dynamic Request Scheduling Optimization in Mobile Edge Computing for IoT Applications. *IEEE Internet of Things Journal, Vol. 7, No. 2, February 2020*.

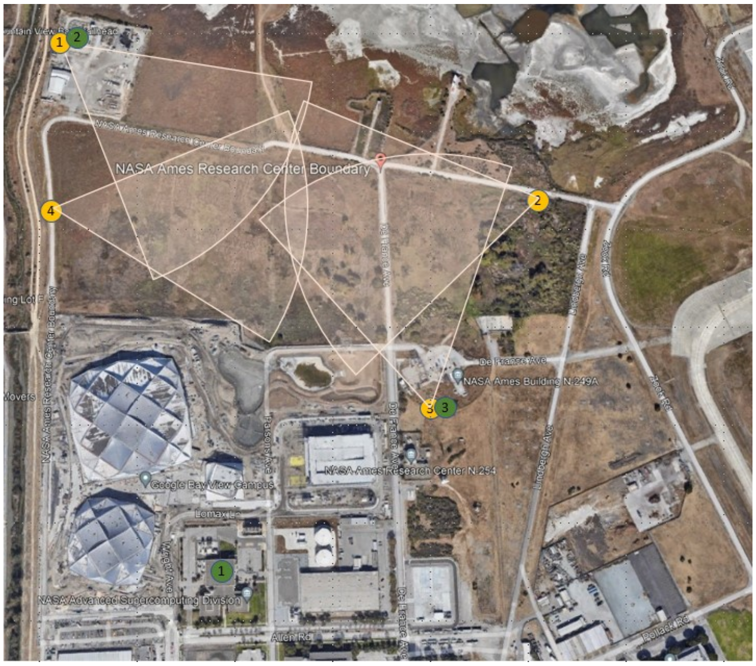
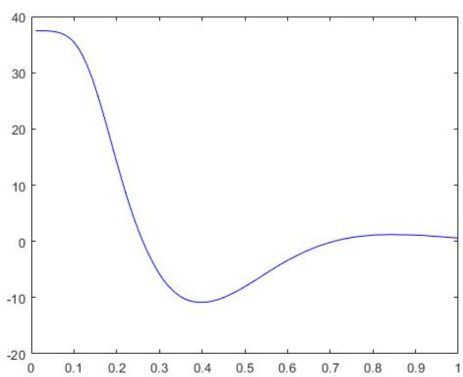
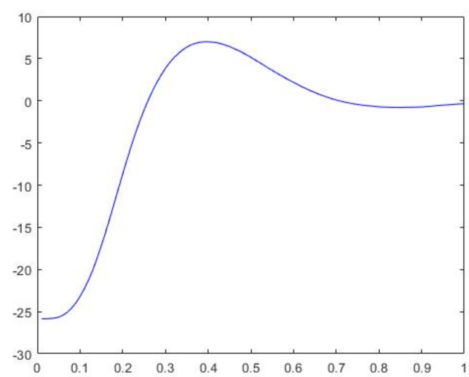


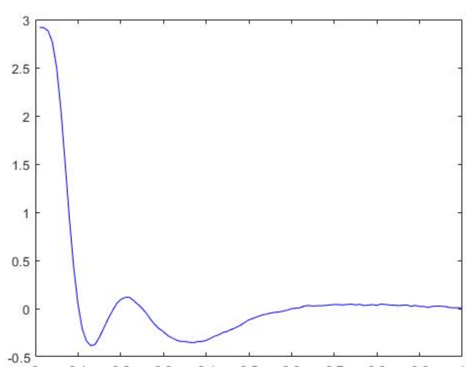
Fig. 2 Sensor's (yellow circles) and computing centers (green circles) locations on site the map.



(a) x-error

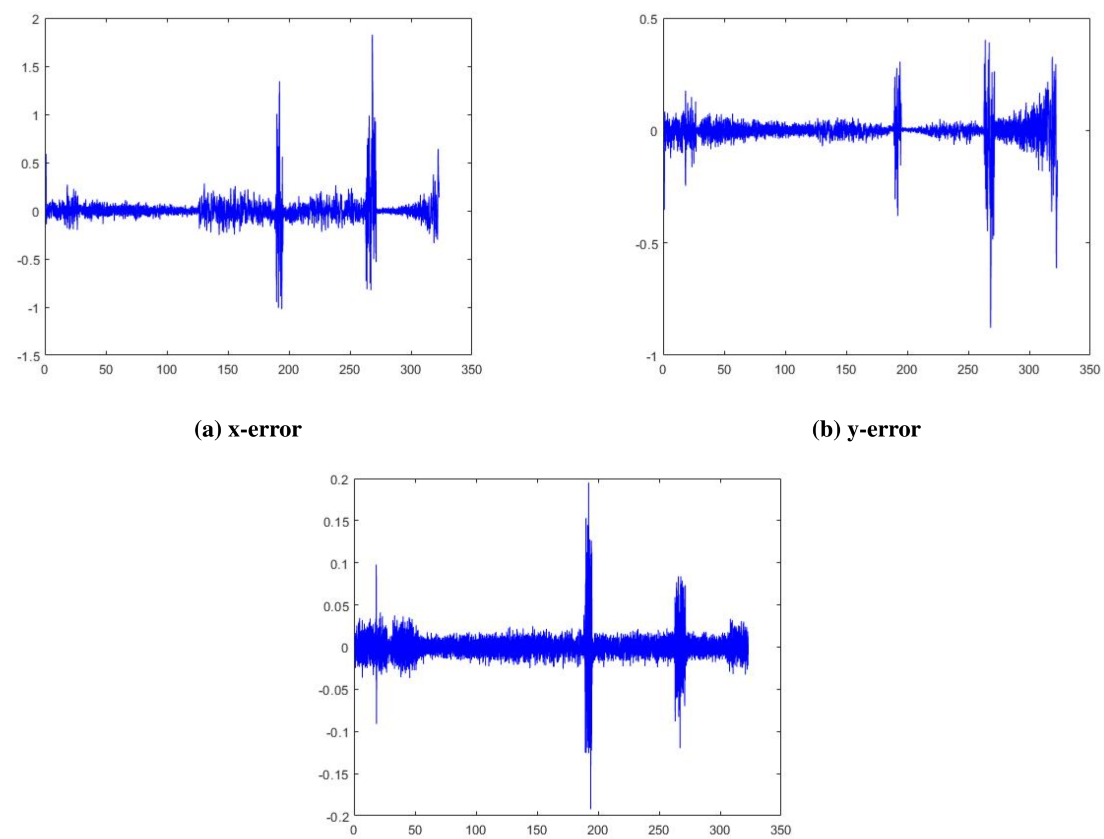


(b) y-error



(c) z-error

Fig. 3 Target coordinates estimation errors for the first second

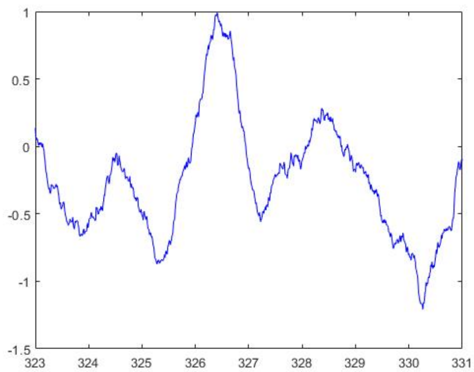


(a) x-error

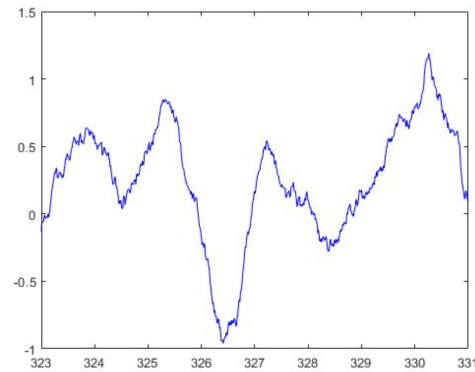
(b) y-error

(c) z-error

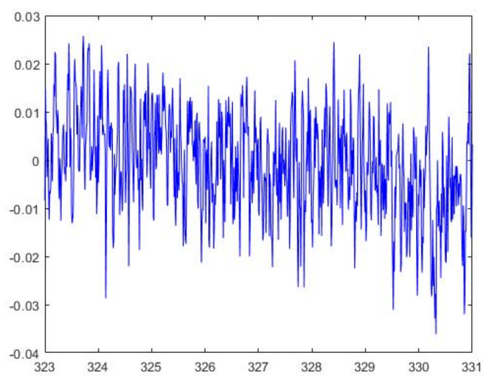
Fig. 4 Target coordinates estimation errors for the during cruising.



(a) x-error



(b) y-error



(c) z-error

Fig. 5 Target coordinates estimation errors during landing

[5] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. Estimation with Applications to Tracking and Navigation: Theory, Algorithms, and Software. *New York: Wiley, 2001*