Fusion of Low-Cost MEMS IMU/GPS Integrated Navigation System
A. D. Abosekeen¹, A. E. Abdalla²

Abstract
In this paper, the multi-sensor data fusion technique based on fuzzy clustering is used to fuse the data from low cost MEMS IMUs to build an INS model. Using this model, the inertial navigation data PVA is extracted. The navigation data PVA is integrated with the GPS data using Kalman filters to build an accurate navigation system of an UAV. Simulation results show that the method can achieve higher accuracy solutions with low cost IMU sensors and improve the performance of integrated navigation system.

Key words
Data Fusion, fuzzy, Data Clustering, Inertial Navigation system (INS), Global Navigation system (GPS), Kalman Filter (KF), and INS/GPS Integration.

1. Introduction
In recent years, obvious attention has been focused on multi-sensor data fusion for military applications like automated target recognition (smart weapons),

¹Master Student, M.T.C, Egyptian Air Forces
²Assoc. Prof, Dean of M.T.C, M.T.C, Egyptian Air Forces
Navigation, guidance for autonomous vehicles, remote sensing, battlefield surveillance, and automated threat recognition systems, such as identification-friend-foe-neutral (IFFN) systems, also for civilian applications as monitoring of manufacturing processes, condition-based maintenance of complex machinery, robotics, and medical applications. Data fusion techniques combine data from multiple sensors and related information to achieve more specific inferences than could be achieved by using a single, independent sensor[1]. Fusion processes can be categorized into three levels of modes (low, Intermediate, and Highlevel fusion), as follows[2]:

1. Lowlevel fusion where several sources of essentially the same type of raw data are combined to produce a new data set that is expected to be more informative and useful than the inputs.
2. Intermediate level fusion or feature level fusion which combines various features into a feature map.
3. Highlevel fusion or decision fusion which combines decisions from several experts. Methods of decision fusion are voting, fuzzy logic, and statistical methods.

The integration between GPS and INS exploits their synergy in various approaches, based on the use of KF, with the goal to mitigate the short time error of GPS and longtime error of INS[3]. The resulting plant is a combined navigation system that has better performance than GPS or INS, considered as stand-alone navigation systems. In order to calculate the estimate of INS error, KF constantly updated by the information from INS and GPS. Using multi-sensor data fusion algorithm to fuse the raw data from several IMUs in INS will improve the integration process between the INS and the GPS systems. In this paper the effect of using multi-sensor data fusion algorithm on the INS/GPS process is studied using aerosonde UAV model see Figure (1)[4]. The reference navigation data (PVA), and gyroscopes (gyros) rates and accelerations are extracted from the aerosonde UAV model.
2. IMU Error Model

There are several sources of INS errors that can be grouped into deterministic and stochastic errors. The deterministic errors include bias offset, scale factor, and axes misalignment errors. The bias offset is a constant error within the sensor measurement. The scale factor is the relationship between the output of the signal of the sensor and the physical quantity being measured. The axes misalignment is the error resulting from the imperfection of mounting the sensors. This misalignment has a deterministic and a stochastic error. The deterministic errors mentioned above can be corrected once they have been properly determined by standard calibration procedures. The stochastic errors include bias drift, bias stability, scale factor stability, noise, and axes misalignment errors. Bias drift can be seen when the measurement drifts from the observable with a relationship that may be linear, quadratic, or of a higher order. The bias stability is the maximum expected drift in a sensor output. This can be measured as a function of time or as a function of temperature. Scale factor stability is the capability of the inertial sensor to accurately sense linear accelerations and angular velocities. Deviations from the theoretical scale are due to system imperfections. The noise results from the sensor itself that interferes with the output signals. Several stochastic models are available for modeling these errors [5].

For simulating IMU measurement, the IMU noise model consists of (gyroscope noise model + accelerometer noise model).
The gyro measurement model is [6, 7]:

$$
\tilde{\omega}_{ib}^b = (I_{3x3} + S_g) \omega_{ib}^b + b_g + \eta_{gy}
$$

(1)

$$
\tilde{b}_g = \eta_{gy}
$$

(2)

Where: $b_g$ is the gyro bias, $S_g$ is a diagonal matrix of gyro scale factors, $\eta_{gy}$ and $\eta_{gu}$ are zero-mean Gaussian white-noise processes with spectral densities given by $\sigma_{gy}^2 I_{3x3}$ and $\sigma_{gu}^2 I_{3x3}$, respectively.

The accelerometer measurement model is [6, 7]:

$$
\tilde{a}^b = (I_{3x3} + S_a) a^b + b_a + \eta_{av}
$$

(3)

$$
\tilde{b}_a = \eta_{au}
$$

(4)

Where: $b_a$ is the accelerometer bias, $S_a$ is a diagonal matrix of accelerometer scale factors $\eta_{av}$, and $\eta_{au}$ are zero-mean Gaussian white-noise processes with spectral densities given by $\sigma_{av}^2 I_{3x3}$ and $\sigma_{au}^2 I_{3x3}$ respectively.

A discrete-time simulation of gyro model using the spectral densities is shown in Error! Reference source not found.).

Where: $N = (0, \sigma^2)$ denotes a zero-mean normal distribution with $\sigma^2$. The same model can be used for the accelerometer[6, 7].
3. Multi-sensor data fusion

Data fusion techniques combine data from multiple sensors and related information to achieve more specific inferences than could be achieved by using a single, independent sensor [1]. Using multi-sensor data fusion in aircraft navigation applications has appeared in recent years with the advent of [low cost, small size and low mass navigation sensors (e.g. optical gyros, MEMS inertial sensors and GNSS sensors)] [8]. In this paper, the multi-sensor data fusion technique is based on fuzzy c-means (FCM).

4. FCM Algorithm

The FCM clustering algorithm is based on the minimization of an objective function called C-means functional. This function can be defined as in Eq.(6) [9-16].

\[
J(X; U, V) = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ik})^{m} \|x_k - v_i\|^2
\]

The FCM algorithm shown in Error! Reference source not found.) computes with the standard Euclidean distance norm, which induces hyperspherical clusters. Hence it can only
detect clusters with the same shape and orientation, because the common choice of norm inducing matrix is $A = I[13]$.

Figure (3) FCM Algorithm

Three parameters in this algorithm have to be determined in the beginning which they are $[c, m, \varepsilon]$, they represent the number of clusters, the weighting parameter that represent the fuzziness of the system, and the ending threshold respectively.

The Algorithm can be explained as follows:

**Step 1** Enter
- Number of clusters $1 < c < N$,
- The weighting exponent $m > 1$,
- The termination tolerance $\varepsilon > 0$.
- Initialize the membership matrix randomly
Repeat for \( l = 1, 2 \ldots \).

**Step 2** Compute Cluster Centers

\[
v^{(l)}_i = \frac{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m x_k}{\sum_{k=1}^{N} (\mu_{ik}^{(l-1)})^m}, 1 \leq i \leq c
\]  
(7)

**Step 3** Compute Distances (Euclidian distance)

\[
D^2_{ik} = (x_k - v_i)^T A (x_k - v_i)
\]  
(8)

**Step 4** Compute the membership matrix

Repeat until

\[
\max |v^{(l)}_i - v^{(l-1)}_i| \leq \varepsilon
\]  
(9)

5. **Fuzzy Clustering as Data Fusion Technique**

The data fusion technique using the fuzzy clustering algorithm is shown in Error! Reference source not found. This technique consists of three main parts:

1. Collecting data from the environment by using multiple sensors.
2. Separating the collected data by using the FCM algorithm.
3. The output from FCM applied to the fusion mechanism to produce a single output.
The first part is concerned with setting the sensors for measuring a particular phenomenon from the environment, the second part uses the FCM to separate the data collected by the sensors to a certain number of clusters with membership matrix and cluster centers, the third part deals with the output clusters and membership functions by generating a binary code, this binary code is representing the membership function of the clusters and generated by comparing the membership function with a certain threshold value, and the minimum binary code denotes to the cluster with the center that represent the fusion process output.

6. Selection of Cluster Number

According to the FCM algorithm, it is required to predefine the number of clusters (c). This number can be calculated with many methods such as the validation parameters but only in offline mode, or by the smooth kernel density estimator (SKDE) which is powerful in the real time and can be obtained as follows [17, 18]:
Where: \( m \) is the length of the estimation window.

The main idea is that the measurements values drifted in two directions around the acceptable region of measurements as shown in Figure (1), so the number of clusters have to be determined every instance of measurement. From the figure below the partitions may be three \((c=3)\) if the drift was in two directions from the accepted region or may be two partitions \((c=2)\) if the drift at any instance were to the left or to the right direction (one direction drift)[19].

![Partitioning for the measurements](image)

Figure (5) partitioning for the measurements

Smooth Kernel density estimator (SKDE) for a given inputs of measurements is shown in Figure (5), where the maximum peak in the figure is denoted as \( Max \) and located by the dotted lines.
Subsequently, the number of clusters is determined according to the following two rules, based on the maximum peak location of the kernel estimator:

1. If the maximum peak of the SKDE is left or right skewed then $C = 2$.
2. If the maximum peak of the SKDE is centered then $C = 3$.

7. Fusion Mechanism

After the clustering process as in Error! Reference source not found.4) each cluster membership function is represented as a binary code $B_{Cl}$. The creation of this code depends upon the membership functions for the clusters and a variable threshold level such that[20].

$$B_{Cl}(s) = \begin{cases} 1, & \text{if } \mu(s) < \alpha \\ 0, & \text{if } \mu(s) > \alpha \end{cases}$$ (11)

Where $(\quad)$ is given as follows:

$$\alpha = 1 - \frac{1}{m} \sum_{j=1}^{c} \sum_{i=1}^{m} \mu_j(s_i) \log \mu_j(s_i)$$ (12)

The first term in the equation above is used for mapping the values, while the second term is an entropy measure for the resulted clustering process from previous section. The fused output will be the cluster center that achieves the following minimization argument for the generated binary codes:
8. Kalman filter

The Kalman filter can be divided into two stages, the time update (prediction), and the measurement update (Correction). In the former as shown in Figure (), the Kalman gain $K_k$ is computed first, and then the state and the error covariance are updated using the prior estimate $\hat{\chi}_k$ and its error covariance $P_k^-$. In the prediction the state is predicted with the dynamic model, and in the correction (update) predicted value is corrected with the observation (measurement) model, so that the error covariance of the estimator is minimized[21, 22].

The different INS error states can be described using the following 1st order state equations:

$$\dot{\chi} = F \chi + G u$$  
(14)

Where: $F$ is the dynamics matrix (state matrix), $\chi$ is the state vector, $u$ is the forcing vector function (input vector), and $G$ is a design matrix (input matrix). These terms are described in details as:

$$\chi = \begin{pmatrix} \delta \chi^p \\ \delta \chi^v \\ \delta \chi^b \end{pmatrix} = \begin{pmatrix} \delta \varphi \\ \delta \lambda \\ \delta \delta N \\ \delta \delta E \\ \delta \delta P \\ \delta \delta L \\ \delta \delta I \\ \delta \delta Z \end{pmatrix}^T$$  
(15)

$$u = \begin{pmatrix} \delta f^b_x \\ \delta f^b_y \\ \delta f^b_z \end{pmatrix} = \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} \delta \omega_x \\ \delta \omega_y \\ \delta \omega_z \end{pmatrix}^T$$  
(16)

$$F = \begin{pmatrix} F_{RR} & F_{RV} & 0_{3x4} \\ F_{VR} & F_{VV} & F_{VAQ} \\ 0_{4x4} & 0_{4x4} & F_{AAQ} \end{pmatrix}, \quad G = \begin{pmatrix} 0_{3x3} & 0_{3x3} \\ G_{AVQ} & 0_{3x3} \\ 0_{4x3} & G_{AAQ} \end{pmatrix}$$  
(17)
Figure (7) Kalman Filter Algorithm

8.1 Lever-arm Correction

Since the sensors cannot be installed at the same place in the host. The position and velocity of the IMU are different from those of the GPS. This is called the lever-arm effect. The lever-arm correction for the position and velocity can be written as [21]

\[
\begin{bmatrix}
\frac{1}{M + h} & 0 & 0 \\
0 & \frac{1}{(N + h) \cos \varphi} & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
\Delta L^b \\
\Delta L^b
\end{bmatrix}
\]

And,

\[
\begin{bmatrix}
\frac{1}{M + h} & 0 & 0 \\
0 & \frac{1}{(N + h) \cos \varphi} & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
\Delta L^b \\
\Delta L^b
\end{bmatrix}
\]

\[
\begin{bmatrix}
\frac{1}{M + h} & 0 & 0 \\
0 & \frac{1}{(N + h) \cos \varphi} & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
\Delta L^b \\
\Delta L^b
\end{bmatrix}
\]
Where:  is the height of moving object,  is the meridian radius in meters,  is the prime vertical radius in meters,  is the offset vector of the GPS antenna or any other IMU from the center of the IMU placed in the CG of the moving object in the body frame,  is the transformation matrix from body to navigation frame, and  is the skew-symmetric matrix of  that represents the projection of the rotating rate vector of the B-frame with respect to the N-frame on the N-frame.

9. The Proposed Scenario and Simulation Results

In this section the steps of the complete process is clarified in the block diagram in Figure (8).

9.1 The Simulation Parameters and Results

1. The GPS
   - Simulation time = 300 seconds.
   - Sampling time = 1 second.
   - Masking angle = 5 degrees.

2. The simulation parameters that make the first IMU measurements be as MEMS grade IMU are as follows:
   - Simulation time  \( T = 300 \) seconds and the sampling time  \( T_s = 0.01 \) seconds.
   - Initial biases for the gyros are  \( b_g = 10 \) deg/hr.
   - Initial biases for accelerometers are  \( b_a = 0.003 \) m/s\(^2\).
   - The gyro noise parameters  \[ \sigma_{g} = 8.7266 \times 10^{-7} \text{ rad/sec}^{1/2} , \sigma_{g'u} = 9.1989 \times 10^{-7} \text{ rad/sec}^{3/2} \].
   - The accelerometer parameters  \[ \sigma_{a} = 1.5 \times 10^{-5} \text{ m/sec}^{3/2} , \sigma_{a'u} = 6 \times 10^{-5} \text{ m/sec}^{5/2} \].
3. The simulation parameters that make the first IMU measurements be as MEMS grade IMU are as follows:
   - Simulation time $T = 300$ seconds and the sampling time $T_s = 0.01$ seconds.
   - Initial biases for the gyros are $b_g = 15$ deg/hr.
   - Initial biases for accelerometers are $b_a = 0.006$ m/s$^2$.
   - The gyro noise parameters
     $\left[ \sigma_{gv} = 7.7266 \times 10^6 \text{ rad/sec}^{1/2}, \sigma_{gu} = 7.1989 \times 10^6 \text{ rad/sec}^{3/2} \right]$.
   - The accelerometer parameters
     $\left[ \sigma_{av} = 2.6 \times 10^{-5} \text{ m/sec}^{3/2}, \sigma_{au} = 7 \times 10^{-5} \text{ m/sec}^{5/2} \right]$.

4. The simulation parameters that make the first IMU measurements be as MEMS grade IMU are as follows:
   - Simulation time $T = 300$ seconds and the sampling time $T_s = 0.01$ seconds.
   - Initial biases for the gyros are $b_g = 20$ deg/hr.
   - Initial biases for accelerometers are $b_a = 0.009$ m/s$^2$.
   - The gyro noise parameters
     $\left[ \sigma_{gv} = 9.7266 \times 10^8 \text{ rad/sec}^{1/2}, \sigma_{gu} = 8.1989 \times 10^8 \text{ rad/sec}^{3/2} \right]$.
   - The accelerometer parameters
     $\left[ \sigma_{av} = 1.9 \times 10^{-5} \text{ m/sec}^{3/2}, \sigma_{au} = 8 \times 10^{-5} \text{ m/sec}^{5/2} \right]$.

5. Assuming that the Lever-arm between the used sensors is as follows:
   - Lever-arm from the first IMU to GPS in body frame: $X = 1.72$ m, $Y = 0.30$ m, $Z = -1.49$ m
   - Lever-arm from the second IMU to GPS in body frame: $X = 1.59$ m, $Y = 0.30$ m, $Z = -1.49$ m
   - Lever-arm from the third IMU to GPS in body frame: $X = 1.99$ m, $Y = 0.30$ m, $Z = -1.49$ m
And the first IMU is assumed is the one placed at the CG of the moving vehicle. The process is also explained at the block diagram in Figure (8).
Figure (8) Block Diagram of Applying the Fusion and the Integration Algorithms
on the Navigation Process of Aerosonde UAV

9.1.1 Reference (P, V, A)

The reference position components in N-frame from the UAV model is shown in Figure (9), the reference velocity components in N-frame is shown in Figure (10), and the reference attitude components is shown in Figure (11).

Figure (9) Reference Position Components in N-frame (Px, Py, Pz)
Figure (10) Reference velocity Components in N-frame (VN, VE, VD)

Figure (11) Reference Attitude Components (Roll, Pitch, Yaw)

9.1.2 Calculation of (P, V, A) RMSE using the INS

a) The Position RMSE
b) The Velocity RMSE

c) The Attitude RMSE
Figure (14) Attitude RMSE

9.1.3 Calculation of (P, V, A) RMSE using the INS/GPS Integration

a) Position RMSE Using Integration without Fusion and with Fusion

Figure (15) Position RMSE of using kalman filter and the GPS and the Fusion with kalman filter
b) Velocity RMSE Using Integration without Fusion and with Fusion

![Velocity RMSE Graphs](image)

Figure (6) Velocity RMSE of using kalman filter and the GPS and the Fusion with kalman filter

c) Attitude RMSE Using Integration without Fusion and with Fusion

![Attitude RMSE Graphs](image)

Figure (7) Attitude RMSE of using kalman filter and the GPS and the Fusion with kalman filter
10. Conclusion

From the results it is clear that the use of stand-alone INS system imply large error growth but when using the integration between the INS and the GPS the error has been decreased, but when the multi-sensor data fusion algorithm is used for fusing the three IMUs measurements, the INS system has been improved and also the integration results also improved by reducing the error of the navigation data (PVA). So from the Results using the multi-sensor data fusion algorithm is improving the navigation process by improving the IMUs measurements, even improve the integration between the INS and the GPS.

References


