2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), Bandung, Indonesia, October 29–30, 2015

Embedded Sensor Fusion and Moving-average Filter for Inertial Measurement Unit (IMU) on the Microcontroller-based Stabilized Platform

Grace Gita Redhyka*, Dika Setiawan, Demi Soetraprawata
Technical Implementation Unit for Instrumentation Development
Indonesia Institute of Sciences
Bandung, Indonesia
*grac002@lipi.go.id

***Abstract*—Sensor Fusion (Complementary and Kalman filters) and Moving Average filter are implemented on an Arduino microcontroller based data acquisition of rotation degree from Inertial Measurement Unit (IMU) sensor for stabilized platform application. Stable platform prototype is designed to have two degrees of freedom, roll and pitch rotation. Output data from gyro and accelerometer were combined to take the advantage of each sensor. Digital filter algorithm was embedded into microcontroller programming. This paper analyzes overshoot percentage, rise time, and data series smoothness of Sensor Fusion (Complementary and Kalman filter) and Moving Average filter response in IMU data acquisition from step input of 20-degreerotation. Moving-average filter resulted in the smallest overshoot percentage of 0% but produce theslowest responsewith 0.42 second rise time. Overall best results are obtained using Complementary filter (alpha value 0.95) by overshoot percentage of 14.17%, 0.24 second rise time, and 0.18 data series smoothness.**

Keywords—Kalman filter; complementary filter; moving average filter; IMU; accelerometer; gyroscope;

I. INTRODUCTION

Stabilized platform is a device that enables object isolation from outside force and keeps it in its inertial state. Inertial stabilized platform have been widely used in diverse applications of engineering, such as stabilizing camera, sensors, and weapon on top of a moving base [1]. The device in this paper was designed to keep the top platform level stable from pitch and roll angular movement. This stabilized platform is utilizing gyro and accelerometer to determine the tilt angle. Traditional mechanical gyro despite having high degree of precision is expensive. In recent years, with therapid development of MEMS technology, IMU has become small, low cost, while promising high-reliability characteristic [2].

Gyro tends to give drifting output since we have to integrate the angular velocity and any offset can accumulate. The accelerometer has jittering characters that can cause system's instability. However this errors can be mitigated through sensor fusion, using the advantage from gyro and accelerometer to correct error from one to another. Gyro drifting error can be compensated by accelerometer's value while the jittering characteristic of the accelerometer is compensated by smoother value of the gyro [3]. Sensor fusion by using Complementary and Kalman filter are applied. Besides that, we also implementMoving-average filter using accelerometer's data only [4-6]. We analyze overshoot percentage, rise time, and standard deviation of the differences on data point (data series smoothness) to determine the performance of each filter.

The goal of this paper is to show that sensor fusion for IMU data acquisition by combining gyroscope and accelerometer using Complementary and Kalman filter can be implemented into themicrocontroller. We also compared them with simpleMoving-average filter of accelerometer. Using the low-cost IMU and open-source Arduino microcontroller, this experiment shows that the system is affordable while maintaining thereliability of the stabilized platform.

II. STABILIZED PLATFORM

A. Mechanical Design

The stabilized platform is designed to have two degrees of freedom rotation, pitch and roll. The platform is made of aluminum with two GWS S777CG/6BB motor servo attached on the pitch frame and roll frame as shown in Fig. 1.

Each motor works as an actuator that move the platform to compensate the pitch or roll rotation from outside force. Both servo motors were controlled by the Arduino board. Both motors are using outside power source from GW Laboratory DC Power Supply GPS-3030.

B. Sensors and Microcontroller

This experiment use LSM303DLHC 3-axis accelerometer and L3GD20 3-axis gyro, that combined into one tiny 0.8" x 0.5" Pololu MinIMU-9 V.2 board. An I²C interface accesses nine independent rotation, acceleration, that can be used to calculate the sensor's absolute orientation. The module includes a voltage regulator and a

978-1-4673-7408-8/15/$31.00©2015 IEEE

level-shifting circuit that allows operation from 2.5 to 5.5 V [7]. The IMU sensor uses 5V power source from Arduino board.

Arduino Mega 2560, a microcontroller board based on the ATmega2560 was used in this experiment. It has SDA (data) and SCL (clock) serial bus that communicate data between the Arduino and sensors board, 5V power compatible for operating the IMU, and digital pulse width modulator (PWM) pin for controlling the servo motors.

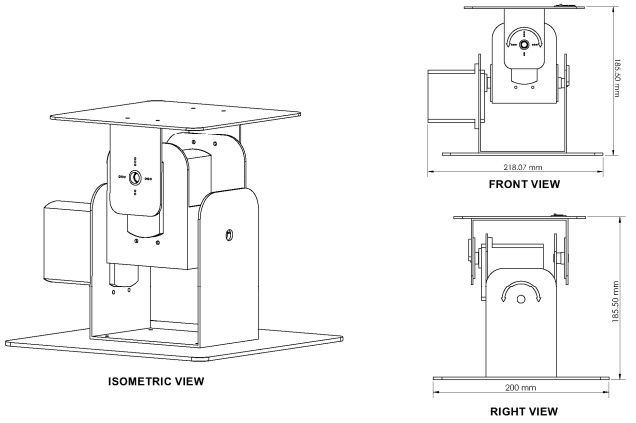

Fig. 1. Drawing of the stabilized platform.

III. METHODS

A. Sensor Fusion and Moving Average

In order to analyze data provided by the IMU, the raw data needed to be converted into angular units. On initialization, average raw sample readings were taken as an offset criterion for each sensor while thesystem is at rest to account for static error, due to temperature sensitivity or misalignment [8].

$$s_{bias} = \frac{\sum_{i=1}^{n} s}{n} \tag{1}$$

Where s is the sensor raw output from initialization and s_{bias} is the offset bias from n samples reading. The s_{bias} value then used to subtract the raw output from sensors.

$$a = \frac{a - a_{bias}}{sensitvity} \tag{2}$$

Accelerometer's raw output a is the sensor's acceleration due to gravity force in units of g, accounting for bias and sensitivity of the sensor. The atan2 function was used to convert units ofginto angular components. Roll angle φ was obtained using the accelerometer value of x axis and z axis, while pitch angle θ was found using the value of$y$ axis and z axis.

$$\varphi_a = \text{atan2}(a_x, a_z) \tag{3}$$

$$\theta_a = \text{atan2}(a_y, a_z) \tag{4}$$

Gyroscope raw values represent angular rates $\dot{\varphi}_g$ and $\dot{\theta}_g$ in degree/second. To obtain angular position of the gyro, integration with respect to time has to be made.

$$\varphi_g = \int_{t_0}^{t} \dot{\varphi}_g \, dt \tag{5}$$

$$\theta_g = \int_{t_0}^{t} \dot{\theta}_g \, dt \tag{6}$$

The definite integral can be approximated by linear approximation, using trapezoid rule[9]. Defining the sampling time step Δt, we can calculate the angular position of the gyroscope, accounting sensor bias and sensitivity.

$$\varphi_g = (\dot{\varphi}_g prev + \dot{\varphi}_g new)\frac{\Delta t}{2} \tag{7}$$

$$\dot{\varphi}_g = \frac{\dot{\varphi}_g - \dot{\varphi}_g bias}{sensitivity} \tag{8}$$

$$\theta_g = (\dot{\theta}_g prev + \dot{\theta}_g new)\frac{\Delta t}{2} \tag{9}$$

$$\dot{\theta}_g = \frac{\dot{\theta}_g - \dot{\theta}_g bias}{sensitivity} \tag{10}$$

Complementary filter was designed using α to determined how much the angle calculation rely on gyroscope or accelerometer.

$$\varphi_c = \alpha(\varphi_c + \varphi_g) + (1-\alpha)\varphi_a \tag{11}$$

$$\theta_c = \alpha(\theta_c + \theta_g) + (1-\alpha)\theta_a \tag{12}$$

Kalman filter was designed by selecting pitch or roll angle as a state vector and then using accelerometer to estimate gyro constant deviation b. The state vector equation drift estimation are as follows [10]:

$$\begin{cases} \hat{x}_k = \mathbf{A}\hat{x}_{k-1} + \mathbf{B}u_k \\ \hat{y}_k = \mathbf{C}\hat{x}_k \end{cases} \tag{13}$$

where $\hat{x}_k$ denotes state vector at k time, $\hat{y}_k$ represents measurement vector, **A** denotes the state space matrix, **B** denotes control matrix, **C**denotesobservation matrix and u_k denotes gyro output containing fixed drift. Using (7) – (10) for IMU state space model, and making accelerometer angle measurement as the current state of measurement vector, (13) can be expanded to:

$$\begin{cases} \begin{bmatrix} \varphi_k \\ b_{gk} \end{bmatrix} = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \varphi_{k-1} \\ b_{gk-1} \end{bmatrix} + \begin{bmatrix} \Delta t \\ 0 \end{bmatrix} \dot{\varphi}_{gk} \\ \hat{y}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \varphi_{ak} \\ b_{gk} \end{bmatrix} \end{cases} \tag{14}$$

where$b$ represents the gyro bias, φ represents the pitch angle, $\dot{\varphi}_{gk}$ represents gyroscope output, and φ_{ak} represents the pitch angle calculated from accelerometer. The discrete-timeKalman filter equations are as follows:

$$\hat{x}_k^- = \mathbf{A}\hat{x}_{k-1} + \mathbf{B}u_k \quad (15)$$

$$\hat{x}_k = \hat{x}_k^- + K(\hat{y}_k - \mathbf{C}\hat{x}_k^-) \quad (16)$$

where the superscript – denotes prior estimate. By inserting (14) to (15) and (16) we can get Kalman equations for the IMU:

$$\varphi_k^- = \varphi_{k-1} + (\dot{\varphi}_k - b_{k-1})\Delta t \quad (17)$$

$$\varphi_k = (1 - K_0)\varphi_k^- + K_0 a_k \quad (18)$$

$$b_{gk} = b_{gk-1} + K_1(\varphi_{ak} - \varphi_{ak}^-) \quad (19)$$

where$K$ is the Kalman gain. TheKalman gain is calculated by first calculating prior estimation of uncertainty prediction matrix $\mathbf{P}^-$ using this equation [10]:

$$\mathbf{P}_k^- = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q} \quad (20)$$

Q is the system process noise covariance matrix. By inserting **A**, the state space matrix, (20) can be expanded to:

$$\begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} \\ \mathbf{P}_{10} & \mathbf{P}_{11} \end{bmatrix}_k^- = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} \\ \mathbf{P}_{10} & \mathbf{P}_{11} \end{bmatrix}_{k-1} \begin{bmatrix} 1 & 0 \\ -\Delta t & 1 \end{bmatrix} + \begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{Q} \end{bmatrix} \Delta t \quad (21)$$

Kalman gain is calculated using the following equation:

$$K_{k,00} = \frac{\mathbf{P}_{00}^-}{\mathbf{P}_{00}^- + \mathbf{R}} \quad (22)$$

where**R** is measurement noise covariance matrix from accelerometer. The updating uncertainty prediction matrix equates to:

$$\mathbf{P}_{00} = (1 - K_{k,00})\mathbf{P}_{00}^- \quad (23)$$

For calculating the roll angle, the same processeswere conducted by choosing the roll angle as the state vector and calculate the prediction using Kalman Filter.

Besides the sensor fusion, a simple Moving-average filter was also implemented. The filter is simply doing average on a moving window of *n* values at any instant. In this experiment, window of 50 values is used. Simple moving-average filter (SMA) is calculated as follows [11]:

$$SMA = \frac{x_k + x_{k-1} + \ldots + x_{k-(n-1)}}{n} \quad (24)$$

$$SMA_{now} = SMA_{previous} + \frac{x_{k-n}}{n} - \frac{x_k}{n} \quad (25)$$

B. Implementation on Arduino Microcontroller

Both sensor fusion and moving-average filter are using the same initialization which consists taking samples for offset criterion, bias compensation, and angle calculation. The accelerometer pitch and roll angle are obtained using atan2 function while the gyroscope pitch and roll angle are obtain using linearized approximation of integral in order to be implemented on microcontroller [12]. Complementary filter is simple and can be easily implemented since it is a linear function of accelerometerand gyroscope. Kalman filter is more complex but the one dimensional version can be implemented on Arduino microcontroller. Moving-average, like Complementary filter, is in a linear form, it is easy to be implemented. All the pseudocode described below is executed in a loop.

1. Thepseudo code of initializationstep is as follow :
   - intializing all variables (timestep, pitchPrediction, etc.)
   - receive raw value from gyroscope and accelerometer, gx, gy, ax, and az
   - for (inti=0; i<100; i++)
     tgx += gx; //total value for 100 sample
     tgy += gy;
     tax += ax;
     tay += ay;
     taz += az;
   - calculate bias value
     bgx = tgx / 100;
     bgy = tgy / 100;
     bax = tax / 100;
     bay = tay / 100;
     baz = (taz / 100) – 256;
   - calculate pitch and roll angle from gyro and accelero
     - pitchAccel = atan2((ay – bay) / sensitivity , (az – baz) / sensitivity) * 180) / 3.14;
     - pitchGyro+= (prevpitchgyro + ((gx – bgx) / sensitivity)) * timestep / 2;
     - rollAccel = (atan2((ax - bax) / sensitivity, (az -baz) / sensitivity) * 180) / 3.14;
     - rollGyro += (prevrollgyro + ((gy – bgy) / sensitivity)) * timestep / 2;

2. The pseudo code of Complementary filter is as follow:
   - compPitch = 0.95*(compPitch + pitchGyro) + 0.05 * pitchAccel;
   - compPitch1 = 0.50 * (compPitch1 +pitchGyro) + 0.50 * pitchAccel;
   - compPitch2 = 0.05 * (compPitch2 +pitchGyro) + 0.95 * pitchAccel;

The same pseudocode algorithm was applied to find the roll rotation by changing pitch component to roll component (pitchGyro and pitchAccel to rollGyro and rollAccel).

3. The pseudocode of Kalman filter is as follow [8]:
   - Calculating estimated uncertain prediction matrix P–:
     - Pxx += timeStep * (2 * Pxv + timeStep * Pvv);
     - Pxv += timeStep * Pvv;
     - Pxx += timeStep * giroVar;
     - Pvv += timeStep * deltaGiroVar;
   - Calculating Kalman gain:
     - kx = Pxx * (1 / (Pxx + accelVar));
     - kv = Pxv * (1 / (Pxx + accelVar));
   - Calculating pitch and roll prediction

- Calculating updated uncertain prediction matrix P:
  - Pxx *= (1 - kx);
  - Pxv *= (1 - kx);
  - Pvv -= kv * Pxv;

4. The pseudocode of Moving Average is as follow [13]:
   - Calculate total reading from accelero sensor
   - Substract the last reading from sensor:
     - total= total - readings[index];
   - Read from the sensor:
     - readings[index] = rollAccel;
   - Add the reading to the total value:
     - total= total + readings[index];
   - Advance to the next position in the array:
     - index = index + 1;
   - Reset index array if arrived at the end of array (50 samples reading):
     - if (index >= numReadings)
       index = 0;
   - Calculate average value by dividing total value by number of readings:
     - average = total / numReadings;

IV. EXPERIMENT

The IMU sensor step response experiment of pitch and roll axis is carried out as follows:

1. Sensor placement

   The step input is simulated by placing the IMU sensor in a motor servo in zero degree rotation and then the servo is actuated to 20^0 position, with adelay of 20ms for pitch axis.

2. Data collection

   IMU data were collected at 1500 data sampling period using step input simulation for Complementary,Kalman filter, and Moving Average respectively and the collected data are saved in thecomputer.

3. Data processing and analysis

   The stored data were processed using MATLAB to display the graph and calculate the overshoot, rise time and graphic smoothness.Overshoot percentage, rise time, and data series smoothness was performed to analyze the performance of each filter algorithm. Overshoot percentage was obtained using [14]:

$$\%OS = \frac{c_{maz} - c_s}{c_s} \times 100\% \qquad (23)$$

where c_{maz} is the maximum value of the step input response, while c_s is steady state point. Rise time criterion is system's response time from 0.1 to 0.9 of the final value desired which is 20^0. Data series smoothness ζ is quickly can be obtained by calculating standard deviation of the difference of each data point $\Delta\varphi_i = \varphi_{i+1} - \varphi_i$, and described as follow [15]:

$$\zeta = \sqrt{\frac{1}{n}(\Delta\varphi_1 - \mu)^2 + ... + (\Delta\varphi_n - \mu)^2} \qquad (24)$$

where $\mu = \frac{1}{n}(\Delta\varphi_1 + ... + \Delta\varphi_n)$, theaverage value of each data point differences. Smaller value of ζ corresponds to smoother data series.

V. RESULTS

As shown in Fig. 2, pitch axis rotation data from accelerometer fluctuates frequently. Data from gyro are much more stable but drifts to zero. Fig. 3 shows Kalman filter response, resulteda quick rise time but with a very high overshoot. The complementary filter was performed using three value of α which resulted the responses as shown by Fig. 4 (α = 0.95), Fig. 5 (α = 0.5), and Fig. 6 (α = 0.05). The figures shows thatvalue of α determined the sensor fusion response characteristic. A higher proportion of gyroscope value will result in more stable result. Moving average algorithm performed using data from accelerometer only is shown by Fig. 7.The comparison of overshoot percentage, rise time and data series smoothness of each filter is summarized on Table 1.

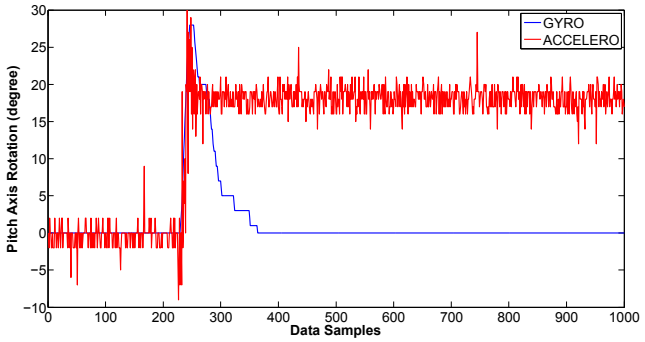

Fig. 2. Pitch axis rotation data output from gyro and accelerometer.

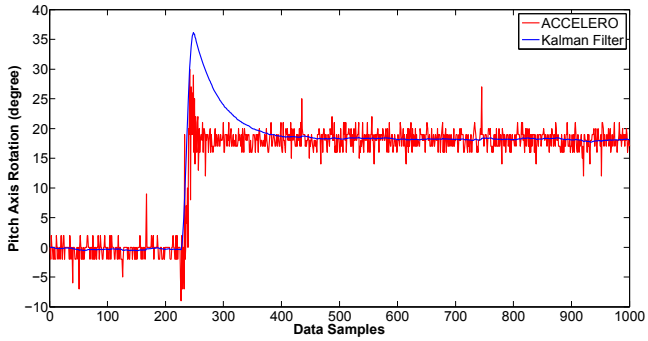

Fig. 3. Pitch axis rotation data output from accelerometer and Kalman Filter.

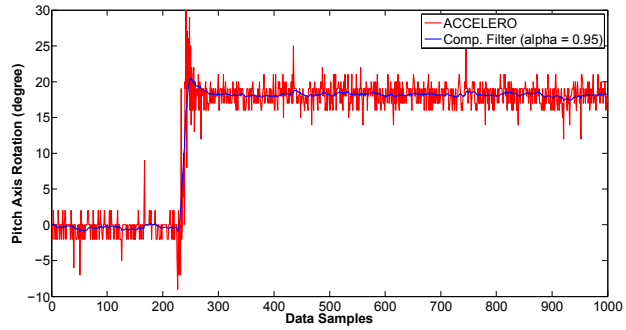

Fig. 4. Pitch axis rotation data output from accelero-meter and Complementary Filter (α = 0.95).

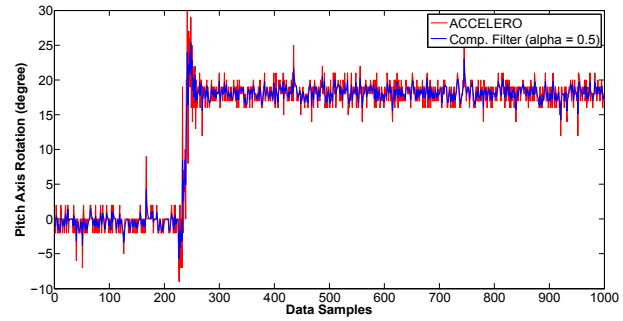

Fig. 5. Pitch axis rotation data output from accelero-meter and Complementary Filter (α = 0.5).

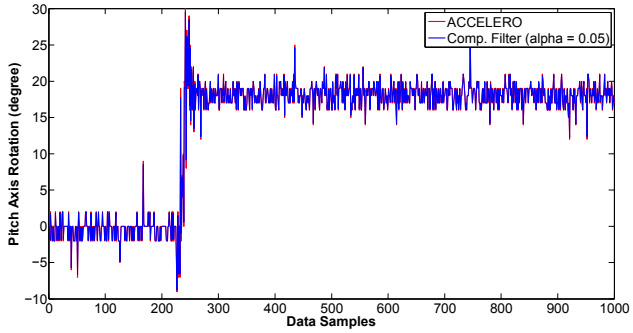

Fig. 6. Pitch axis rotation data output from accelero-meter and Complementary Filter (α = 0.05).

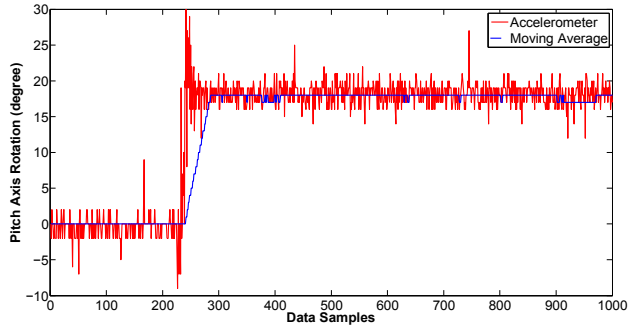

Fig. 7. Pitch axis rotation data output from accelero-meter and Moving Average.

TABLE I. PERFORMANCE COMPARISON OF EACH FILTER

Filter	Result		
	Overshoot	*Rise time (s)*	*Smoothness*
Complementary Filter α = 0.95	14.17 %	0.24	0.18
Complementary Filter α = 0.5	41 %	0.2	1.14
Complementary Filter α = 0.05	64 %	0.2	2.66
Kalman Filter	107.22 %	0.16	0.29
Moving Average	0 %	0.42	0.21

A high overshoot in Kalman filter resulted from accelerometer's data. The pitch prediction value is calculated using accelerometer's pitch which even though did not drift, it fluctuates over time. A better value for accelerometer variance is needed to get optimum Kalman gain. The pitch rotation value resulted from complementary filter shows a smaller overshoot than Kalman filter. Smaller overshoot is important to make the platform system more stable. Moving average filter resulted 0% overshoot but lack of responsiveness with a slow rise time. A quick rise time is needed to make the system responsive to the disturbance.

VI. CONCLUSION

In this paper, an implementation of complementary, Kalman, and moving-average filter to embedded system has been developed. Fig. 2 shows the characteristics of each sensor. Gyroscope sensor produces a smooth value but drifts to zero while the accelerometer eventhough not shifted, its values fluctuate. Sensor fusion using complementary and Kalman filter was implemented on themicrocontroller to improve the IMU readings.Overall, the complementary filter using α = 0.95resulted inbetter system for stable platform application, which needs a quick respond to rotation movement, smaller overshoot, and smoother motor movement.

ACKNOWLEDGMENT

This research was supported by the thematic program (No.3425.001.017) through the Bandung Technical Implementation Unit for Instrumentation Development, Deputy for ScientificServices, funded by Indonesian Institute of Sciences,Indonesia.

REFERENCES

[1] A. R. Alias, M. S. Alias, I. Z. Shamsuddin, R. A. R. Ahmad, and S. N. H. S. Abdullah, "Measure the Ability and Limitation of Gyroscope, Acceleration and Gyro-accelaration for Stabilized Platform," in *Intelligent Robotics Systems: Inspiring the NEXT*, Springer, 2013, pp. 405–415.

[2] G. Li, Y. He, Y. Wei, S. Zhu, and Y. Cao, "The MEMS gyro stabilized platform design based on Kalman Filter," in Optoelectronics and Microelectronics (ICOM), 2013 International Conference on, 2013, pp. 14–17.

[3] C. W. Kang and C. G. Park, "Attitude estimation with accelerometers and gyros using fuzzy tuned Kalman filter," in Control Conference (ECC), 2009 European, 2009, pp. 3713–3718.

[4] K. Abdulrahim, C. Hide, T. Moore, and C. Hill, "Aiding MEMS IMU with building heading for indoor pedestrian navigation," in

Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010, 2010, pp. 1–6.

[5] D. Hazry and others, “A simple approach on implementing IMU sensor fusion in PID controller for stabilizing quadrotor flight control,” in Signal Processing and its Applications (CSPA), 2011 IEEE 7th International Colloquium on, 2011, pp. 28–32.

[6] M. Kamil, T. Chobtrong, E. Günes, and M. Haid, “Low-cost object tracking with MEMS sensors, Kalman filtering and simplified two-filter-smoothing,” Applied Mathematics and Computation, vol. 235, pp. 323–331, 2014.

[7] “Pololu - MinIMU-9 v2 Gyro, Accelerometer, and Compass (L3GD20 and LSM303DLHC Carrier).” [Online]. Available: https://www.pololu.com/product/1268/resources. [Accessed: 26-Jun-2015].

[8] B. McCarron, “Low-Cost IMU Implementation via Sensor Fusion Algorithms in the Arduino Environment,” California Polytechnic State University, San Luis Obispo, 2013.

[9] K. E. Atkinson, *An introduction to numerical analysis*. John Wiley & Sons, 2008.

[10] G. Welch and G. Bishop, “An introduction to the kalman filter. 2006,” University of North Carolina: Chapel Hill, North Carolina, US, 2006.

[11] Y.-L. Chou, Statistical Analysis, 2nd edition. New York: Holt,Rinehart & Winston of Canada Ltd, 1975.

[12] H. Ferdinando, H. Khoswanto, and D. Purwanto, “Embedded Kalman filter for inertial measurement unit (IMU) on the ATMega8535,” in Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on, 2012, pp. 1–5.

[13] “Arduino, Examples, Analog I/O - Smoothing.” [Online]. Available: https://www.arduino.cc/en/Tutorial/Smoothing. [Accessed: 26-Jun-2015].

[14] N. S. Nise, CONTROL SYSTEMS ENGINEERING, (With CD). John Wiley & Sons, 2007.

[15] R. Barnes, “Variogram tutorial,” Golden, CO: Golden. Software. Available online at http://www. goldensoftware. com/. variogramTutorial. pdf, 2003.

[16] W. Li and J. Wang, “Effective adaptive Kalman filter for MEMS-IMU/magnetometers integrated attitude and heading reference systems,” Journal of Navigation, vol. 66, no. 01, pp. 99–113, 2013.

[17] A. Abdelgawad and M. Bayoumi, “Low-power distributed Kalman filter for wireless sensor networks,” EURASIP Journal on Embedded Systems, vol. 2011, p. 3, 2011.