A Dual Extended Kalman Filter for Tilt Estimation

Herman I. Veriñaz, Edgar M. Vela, *Member, IAENG*, Ronald A. Ponguillo, *Member, IAENG*

***Abstract*— Accelerometers and Gyroscopes are typically used to measure rotation angles. However, the nature of both sensors makes difficult to estimate an angle using only one of these sensors. This is because gyroscopes return high bias data, and, since accelerometers detects every small acceleration in each axis, the output of this sensor is very noisy.**

A common solution is to combine both signals in so-called sensor fusion. Two popular techniques are Complementary Filters and Constant Gyro Bias Kalman Filters. These algorithms are attractive because they are simple to implement and do not depend on specific parameters of the system.

Because these filters use the arctangent function, they cannot resolve a discontinuity on the estimated signal. This discontinuity is caused by a flip among the two endpoints in the range of arctangent function. This range is usually [-π/2, π/2] or [-π, π]. This problem occurs because of the discontinuous nature of the tangent function, and because tangent is strictly not invertible. To solve this, an additional routine must be implemented to patch these flips.

This paper presents a practical Dual Extended Kalman Filter algorithm for angle estimation. This work focus on the restricted problem of measuring the angle of rotation of a body respect to one axis parallel to the earth surface. The main characteristic of the developed algorithm is that it does not depend on physical parameters and does not use the inverse tangent function on its implementation.

For the implementation, the accelerometer and gyro signals were acquired from the IMU MPU-6050 with a 50 ms sampling time. The complete algorithm was implemented in a MATLAB script and then it was compared with two other methods usually used in tilt estimation: Complementary Filters and Constant Gyro Bias Kalman Filter.

***Index Terms*—Angle Estimation, Dual Extended Kalman Filter, Sensor Fusion, Kalman Filter, Tilt Estimation.**

I. Introduction

In many applications it is needed to know the orientation of a body respect to a certain coordinate system. This work focus on the restricted problem of measuring the angle of rotation of a body respect to one axis parallel to the earth surface.

There are two common sensors used to measure this orientation: accelerometers, and gyroscopes. One way of doing this is to use an accelerometer to find the gravity components, then the angle is calculated using an inverse tangent function. Other form is to use a gyroscope to measure the angular speed, and the angle is simply this angular speed accumulated over the time.

Both techniques have their drawbacks. The accelerometer does not measure only the gravity components. It also measures every small acceleration in each axis, so its signals are usually noisy. On the other hand, the gyroscope is less noisy but the accumulation of the angular speed over the time also causes an accumulation of the noise, producing a bias in the angle signal.

To overcome these problems two popular techniques of sensor fusion are usually used: Complementary filters and Kalman filters [2] [4].

A complementary filter is employed because it is very practical, and has low computational complexity. It is a weighted sum of the gyro and accelerometer data:

$$\theta_{k+1} = a(\theta_k + \omega_k) + (1-a)z_k \qquad (1)$$
$$z_k = \arctan(a_{mt_k}/a_{mr_k})$$

a: Adjustable parameter between 0 and 1.
z_k: Measured angle at the k_{th} instant.
a_{mt_k}: Measured tangential acceleration at the k_{th} instant.
a_{mr_k}: Measured radial acceleration at the k_{th} instant.
θ_k: Angle of rotation at the k_{th} instant.
ω_k: Angular velocity at the k_{th} instant.

Also, it is typically preferred a Linear Kalman filter [2] using the following constant gyro bias model:

$$\theta_{k+1} = \theta_k + (b_k + \omega_k) \qquad (2)$$
$$b_{k+1} = b_k$$

b_k: Gyroscope bias at the k_{th} instant.

Both filters are simple to implement and do not depend on physical parameters.

This paper presents a Dual Extended Kalman Filter [1] algorithm designed for tilt estimation. The algorithm relies principally on the gyroscope measures that are less noisy than accelerometer signals. This model does not depend on

Manuscript received March 20, 2017; revised March 28, 2017.

H. I. Veriñaz is with the Escuela Superior Politécnica del Litoral, ESPOL, Faculty of Natural Science and Mathematics, Campus Gustavo Galindo, Km 30.5 Via Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador (e-mail: hverinaz@espol.edu.ec).

E. M. Vela is an engineer in Electronics and Telecommunications, graduated at Escuela Superior Politécnica del Litoral, ESPOL, Faculty of Electrical and Computer Engineering, Campus Gustavo Galindo, Km 30.5 Via Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador (e-mail: emvela@espol.edu.ec).

R. A. Ponguillo is the Coordinator of the Basic Electronics Area and a Researcher of the Vision and Robotics Center at the Escuela Superior Politécnica del Litoral, ESPOL, Faculty of Electrical and Computer Engineering, Campus Gustavo Galindo, Km 30.5 Via Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador (e-mail: rponguil@espol.edu.ec).

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

physical parameters, does not use the arctangent function and it was designed and tested using MATLAB software.

II. Mathematical Model

As mentioned in the introduction, this model greatly relies on the angular speed measured with the gyroscope. Ideally, the measured angle is simply the accumulated angular speed over the time. If the sampling time is one, the equation is:

$$\theta_{k+1} = \theta_k + \omega_k \quad (3)$$

θ_k: Measured angle at the k_{th} instant.
ω_k: Angular velocity at the k_{th} instant.

To improve the model, additional information is obtained from the accelerometer. The accelerometer measures the gravity and the acceleration at the same time. This is represented as:

$$a_{mt_k} = a_{t_k} + gcos(\theta_k) \quad (4)$$
$$a_{mr_k} = a_{r_k} + gsin(\theta_k)$$

a_{t_k}: Tangential acceleration at the k_{th} instant.
a_{r_k}: Radial acceleration at the k_{th} instant.
a_{mt_k}: Measured tangential acceleration at the k_{th} instant.
a_{mr_k}: Measured radial acceleration at the k_{th} instant.
g: Local gravity.

The tangential and radial acceleration are:

$$a_{t_k} = r\alpha_k + a_{tk_{cm}} \quad (5)$$
$$a_{mr_k} = -r{\omega_k}^2 + a_{rk_{cm}} \quad (6)$$

α_k: Angular acceleration respect to the center of curvature at the k_{th} instant.
r: Radius of curvature.
$a_{tk_{cm}}$: Tangential acceleration of the center of mass relative to an inertial frame.
$a_{rk_{cm}}$: Radial acceleration of the center of mass relative to an inertial frame.

The terms $a_{tk_{cm}}$ and $a_{rk_{cm}}$ can be used to give additional information to the system. For example, if it is plausible, an additional accelerometer can be placed in the center of mass. On this work, a null acceleration model is used for the center of mass. Thus, the center of mass is the inertial frame.

The radius of curvature is a parameter needed for the implementation. In specific applications, this parameter can be measured directly, but for generality and practicability of the filter, it is assumed unknown. As there is not much information about this parameter, therefore a simple model is used:

$$r_{k+1} = r_k \quad (7)$$

Even though the radius is assumed to be constant, the appropriate tuning of the Kalman filter will allow this estimated value to change over the time.

Thus, the complete model is:

$$\theta_{k+1} = \theta_k + \omega_k \quad (8)$$
$$r_{k+1} = r_k$$
$$z_k = h(r_k, \theta_k, \alpha_k, \omega_k) = \begin{pmatrix} r_k\alpha_k + gcos(\theta_k) \\ -r_k\omega_k^2 + gsin(\theta_k) \end{pmatrix}$$

α_k: Angular acceleration at the k_{th} instant.
r_k: Radius of curvature.
g: Local gravity.
θ_k: Angle of rotation at the k_{th} instant.
ω_k: Angular velocity at the k_{th} instant.
z_k: Observation vector

III. Kalman Filter

The algorithm for the implementation is a Dual Extended Kalman Filter [1] (DEKF). A linear Kalman filter [2] and an extended Kalman filter are executed at the same time, for the radius estimation and angle estimation respectively. The Kalman equations are:

Prediction:

$$\hat{\theta}_{k|k-1} = \hat{\theta}_{k-1|k-1} + \omega_k \quad (9)$$
$$P_{\theta_{k|k-1}} = P_{\theta_{k-1|k-1}} + Q_{\theta_k}$$
$$\hat{r}_{k|k-1} = \hat{r}_{k-1|k-1}$$
$$P_{r_{k|k-1}} = P_{r_{k-1|k-1}} + Q_{r_k}$$

Update:

$$Y_k = z_k - h(\hat{r}_{k|k-1}, \hat{\theta}_{k|k-1}, \alpha_k, \omega_k) \quad (10)$$
$$K_{\theta_k} = P_{\theta_{k|k-1}} H_{\theta_k}^T \left(H_{\theta_k} P_{\theta_{k|k-1}} H_{\theta_k}^T + R_k\right)^{-1} \quad (11)$$
$$\hat{\theta}_{k|k} = \hat{\theta}_{k|k-1} + K_k Y_k \quad (12)$$
$$P_{\theta_{k|k}} = (I - K_{\theta_k} H_{\theta_k}) P_{\theta_{k|k-1}} \quad (13)$$

$$K_{r_k} = P_{r_{k|k-1}} H_{r_k}^T \left(H_{r_k} P_{r_{k|k-1}} H_{r_k}^T + R_k\right)^{-1} \quad (14)$$
$$\hat{r}_{k|k} = \hat{r}_{k|k-1} + K_k Y_k \quad (15)$$
$$P_{\theta_{k|k}} = (I - K_{r_k} H_{r_k}) P_{r_{k|k-1}} \quad (16)$$

The observation model matrix H, for each state variable is the gradient of h, which is the observation model function.

$$H_{r_k} = \begin{pmatrix} \alpha_k \\ -\omega_k^2 \end{pmatrix}$$

For the angle, the gradient matrix is:

$$H_{\theta_k} = \begin{pmatrix} -gsin(\hat{\theta}_{k|k-1}) \\ gcos(\hat{\theta}_{k|k-1}) \end{pmatrix}$$

The vector z_k is the pair of measured accelerations:

$$z_k = \begin{pmatrix} a_{mt_k} \\ a_{mr_k} \end{pmatrix}$$

Finally, notice that matrix R_k is symmetric because signals are real, and Q matrices are simply real numbers:

$$R_k \in S_{2x2}$$
$$Q_{\theta_k}, Q_{r_k} \in \mathbb{R}$$

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

These equations can be implemented in an IDE that handle matrices, like MATLAB. However, the idea is to obtain one-dimensional equations that could be easily implemented in a microcontroller. To simplify the problem, it was assumed that the observation noise is uncorrelated. This means the matrix $R_k$is diagonal.

Notice that the prediction equations (9) are one-dimensional equations and (10) can be write as two simple one-dimensional equations.

Now, to compute the gain matrix K_{θ_k} it is needed to calculate a determinant result of the inverse in equation (11), the equation is:

$$det_\theta = g^2 p_{\theta_{k|k-1}}\left((r_{11} - r_{22})\, cos^2\left(\hat{\theta}_{k|k-1}\right) + r_{22}\right) + r_{11} r_{22} \quad (17)$$

Then, the two components of the matrix K_{θ_k} can be obtained using equation (11), these are:

$$K_{\theta_{1k}} = -\frac{g p_{\theta_{k|k-1}} sin(\hat{\theta}_{k|k-1}) r_{22}}{det_\theta} \quad (18)$$

$$K_{\theta_{2k}} = \frac{g p_{\theta_{k|k-1}} cos(\hat{\theta}_{k|k-1}) r_{11}}{det_\theta} \quad (19)$$

To compute the equations (12) and (13) it is needed to do some matrix algebra, the reduced expressions are:

$$\hat{\theta}_{k|k} = \hat{\theta}_{k|k-1} + K_{\theta_{1k}} Y_{k1} + K_{\theta_{2k}} Y_{k2} \quad (20)$$

$$p_{\theta_{k|k}} = p_{\theta_{k|k-1}}(g K_{\theta_{1k}} sin(\theta_k) - g K_{\theta_{2k}} cos(\theta_k) + 1) \quad (21)$$

In the same way, the equations for the radius are:

$$det_r = p_{r_{k|k-1}} r_{11} \omega_k^4 + p_{r_{k|k-1}} r_{22} \alpha_k^2 + r_{11} r_{22} \quad (22)$$

$$K_{r_{1k}} = \frac{\alpha_k p_{r_{k|k-1}} r_{22}}{det_r} \quad (23)$$

$$K_{r_{2k}} = -\frac{\omega_k^2 p_{r_{k|k-1}} r_{11}}{det_r} \quad (24)$$

$$\hat{r}_{k|k} = \hat{r}_{k|k-1} + K_{r_{1k}} Y_{k1} + K_{r_{2k}} Y_{k2} \quad (25)$$

$$p_{r_{k|k}} = p_{r_{k|k-1}}(-K_{r_{1k}} \alpha_k + K_{r_{2k}} \omega_k^2 + 1) \quad (26)$$

Once these equations are obtained, it is needed to define the values of the covariance matrices R_k,$Q_{\theta_k}$and Q_{r_k}.There is no standard method to obtain these matrices. On this implementation, they were tuned manually by trial and error.

IV. Implementation and Results

For implementation, the MPU-6050 [3] was employed. This is a popular device among hobbyist. It is a MEMS accelerometer and a MEMS gyro in a single chip. For this implementation, both sensors were sampled each 50 ms.

The algorithm was tested using a MATLAB script. The equations implemented were the one-dimensional prediction equations (9), equation (10) and (17) to (26). Matrices Q and R obtained after the tuning are:

$$R_k = \begin{pmatrix} 250 & 0 \\ 0 & 250 \end{pmatrix}$$

$$Q_{\theta_k} = 0.000021$$

$$Q_{r_k} = 20$$

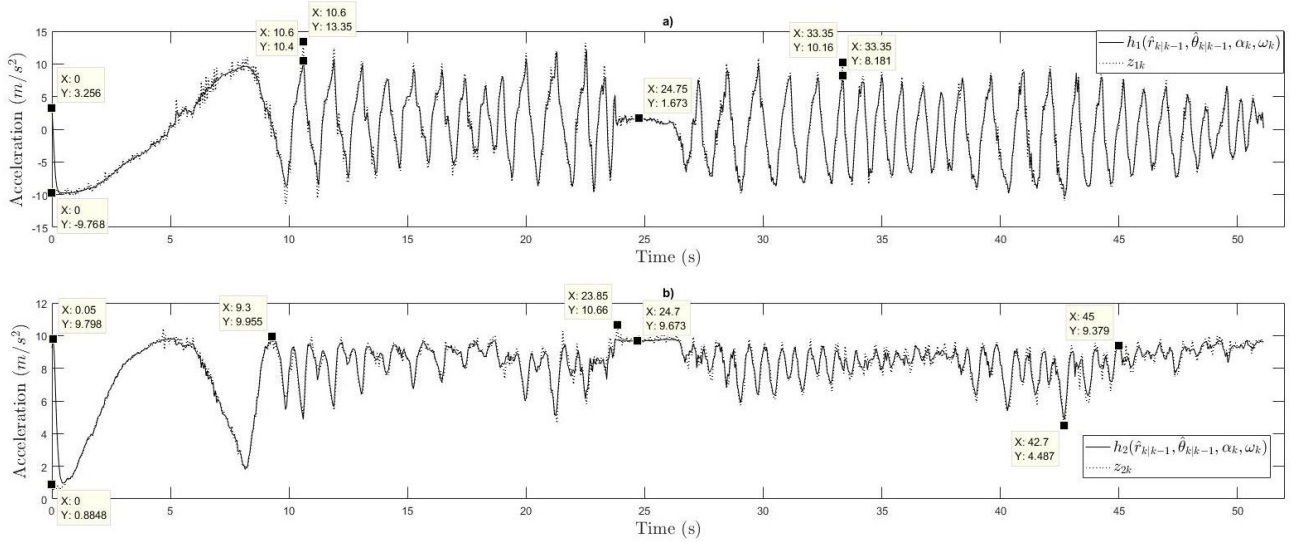

Fig. 1. Accelerometer input (Solid) and estimated acceleration read (Dotted):
a) Tangent acceleration b) Radial acceleration.

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

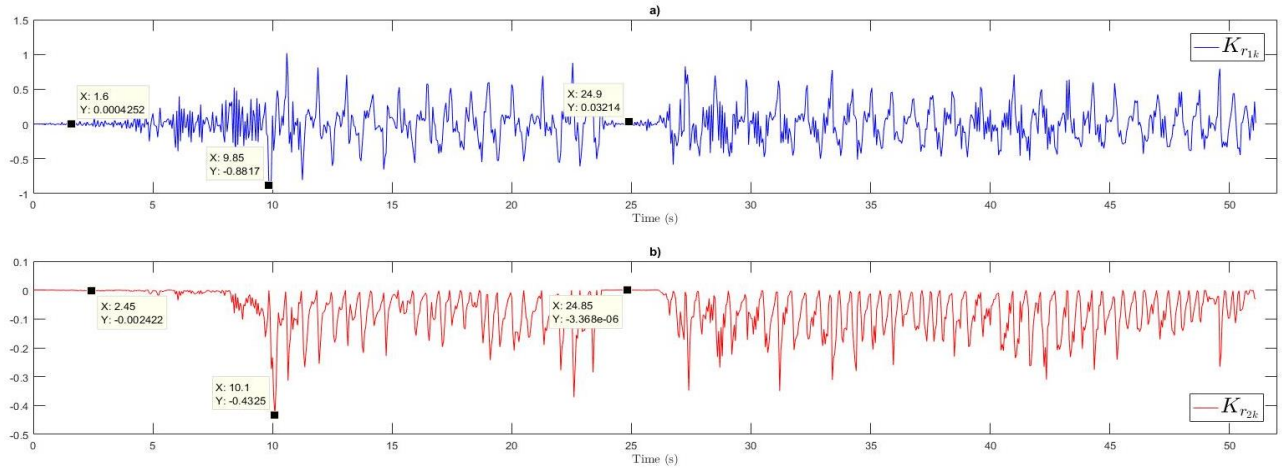

Fig. 2. Gain matrix elements for the estimated radius.
a) First component. b) Second component.

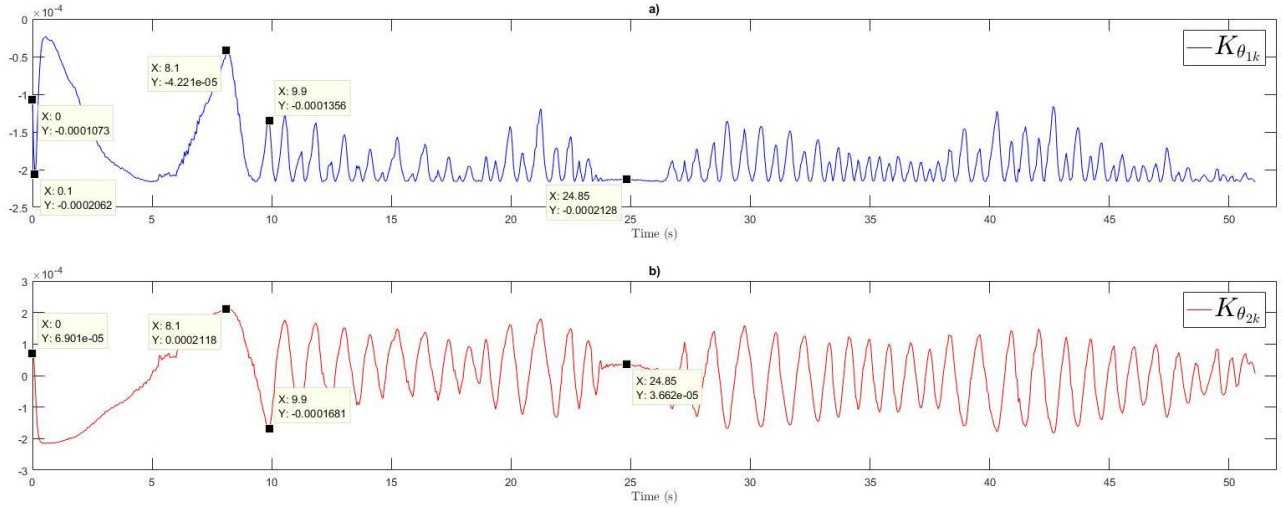

Fig. 3. Gain matrix elements for the estimated angle.
a) First component. b) Second component.

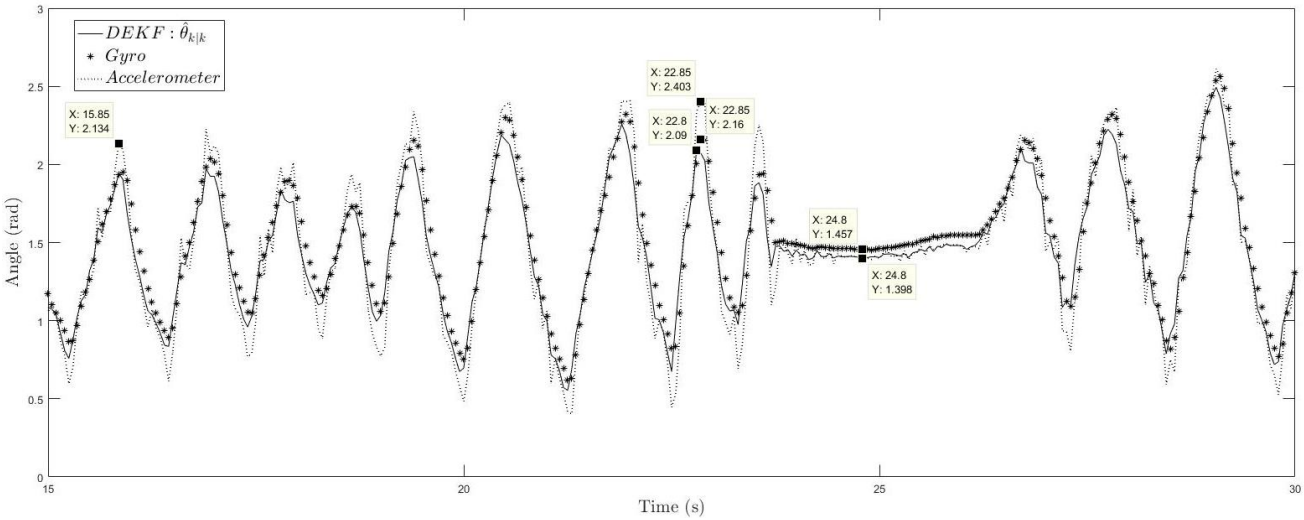

Fig. 4. Tilt angle measure using directly the Gyroscope reading, the Accelerometer reading, and the estimation using DEKF

An adequate tuning allows the predicted signals to follow the sensor signal. This is shown in Figure 1. The estimated signal does not have the same initial value than the read signals, but after approximately 0.5 seconds the filter converges, and both signals are approximately equal.

Notice, that while the filter output converged after 0.5 seconds, the gain matrices did not converge. The gain matrix K_{r_k} for the radius is shown in the Figure 2, and the gain matrix K_{θ_k} for the angle is shown in the Figure 3. The elements of these matrices do not converge to any value, rather they are always being self-adjusted during the whole simulation.

The angle estimated by the filter is shown in Figure 4, and it is compared with the angle calculated integrating the gyro

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

signal, and with the angle calculated with the data acquired from the accelerometer.

The estimated radius is shown in Figure 5, as mentioned before, this signal is not constant even though a constant model was assumed.

V. Conclusions

During twenty seconds, an encoder measure was compared with the CF (1), LKF (2) and DEKF outputs. The Euclidean distance between the encoder signal and the signal of each filter was used as a similarity measure. All filters variables were initialized to zero and, to test the filters response, the tangential acceleration input was intentionally fixed to zero at time 9.15 seconds. For the same reason, at 15.9 seconds the gyro input was fixed to minus one. This is shown in Figure 6.

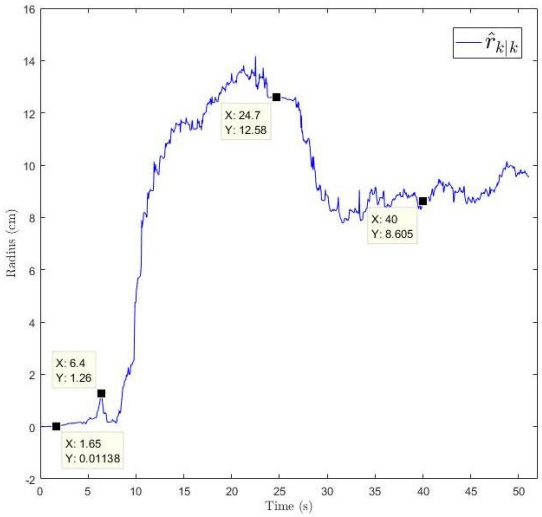

Fig. 5. Estimated Radius using the DEKF

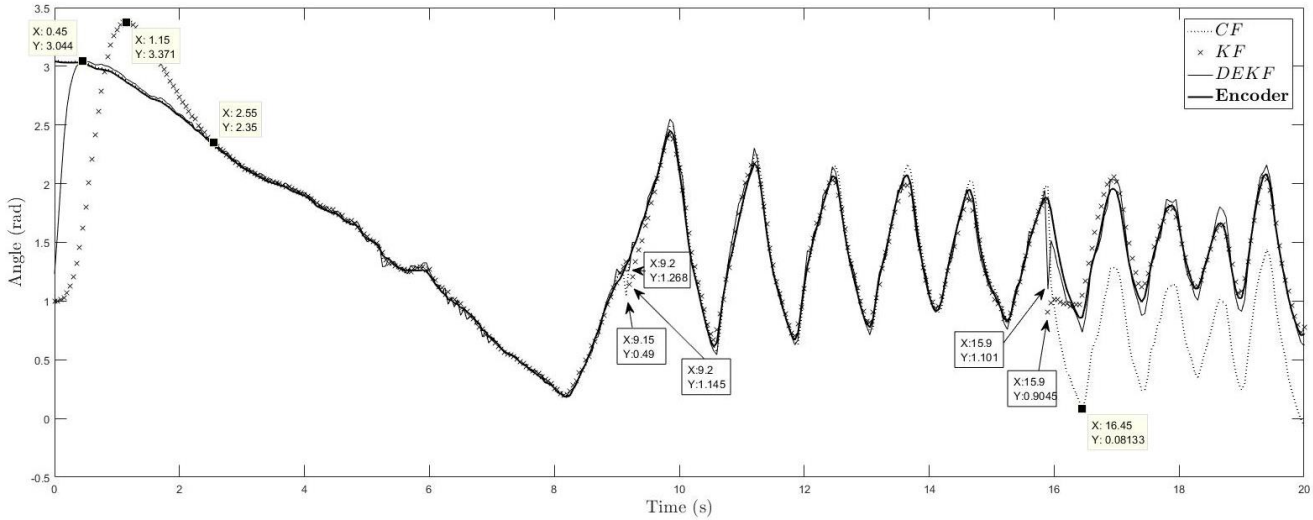

Fig. 6. Encoder measured angle compared with three filters: Complementary, Linear Kalman, and Dual Extended Kalman

The CF (1) was tune with a=0.8. This filter is the more distant to the encoder angle signal (d=20.0594 radians), but it is simpler to implement. Moreover, because it uses directly a weighted average between the gyro and accelerometer signals, the time of convergence is zero. For the same reason, at time 15.9 s the complementary filter cannot restore its estimated signal, but at time 9.15 s it was successfully restored.

The LKF model (2) is simpler than the DEKF, but its time of response is longer. This filter can successfully restore its value at 9.15 s and 15.9 s, and it has a distance to the encoder signal of d= 6.8386 radians. Nevertheless, when distances were measured between 9.85 and 14.65 seconds, these values were d=0.6047 radians, d=0.5105 radians and d=0.4805 radians, for the CF, LKF, DEKF respectively. It means that, if there is no perturbation, and all the filters have converged, the LKF estimation is closer to the encoder reading than the DEKF estimation.

The distance between the encoder signal and the DEKF signal is d=3.2505 radians, the shortest one. This is because this filter converges faster than the LKF and it has a better response to input perturbations, as shown in Figure 6.

The CF and the LKF stand out because of the simplicity of the model. However, it is important to note that, to estimate angles out of the range of the arctangent function, it is needed to implement an additional routine to make the tangent continuous in the desired interval. The DEKF algorithm use directly the sine and cosine functions, so it does not need any additional routine.

References

[1] Nejad, S. and Gladwin, D.T. and Stone, D.A., "On-chip implementation of Extended Kalman Filter for adaptive battery states monitoring," IECON Proceedings (Industrial Electronics Conference) 2016, pp. 5513-5518

[2] Jadan, H.I.V. and Vera, C.R.M. and Intriago, R.P. and Padilla, V.S., "Implementing a Kalman filter on FPGA embedded processor for speed control of a DC motor using low resolution incremental encoders," Lecture Notes in Engineering and Computer Science: Proceedings of World Congress on Engineering and Computer Science, WCECS 2016, Vol.1, pp.367-371

[3] InvenSense, "MPU-6000 and MPU-6050 Product Specification Revision 3.4", MPU600 Datasheet, Nov. 2010 [Revised Aug. 2013]

[4] Veriñaz, H.and Martínez, R.and Ponguillo, R. and Padilla V. S., “Deployment of a Competition Sprinter Robot over FPGA Platform with Feedback Control Systems for Velocity and Position,” Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2017, Vol. 2, pp. 660-665

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)