Sensor Fusion of INS, Odometer and GPS for Robot Localization

Sofia Yousuf, Muhammad Bilal Kadri
College of Engineering
Karachi Institute of Economics and Technology (PAF-KIET)
Karachi, Pakistan
sofiayousuf4@gmail.com, bilal.kadri@pafkiet.edu.pk

***Abstract*—This paper presents data fusion of three sensors Inertial Navigation System (INS), Global positioning systems (GPS) and odometer for determining the correct location of a differential drive mobile robot. The data from INS and odometer is combined using Kalman Filter (KF) based sensor fusion technique. The KF filtered signal and the GPS signal is fused by assigning weights. The proposed technique is tested in simulation. Mathematical models of the three sensors as well as the robot model is developed in MATLAB/Simulink environment to generate the data for simulation purpose. It has been demonstrated that with the proposed sensor fusion architecture exact geo-location of a differential drive wheeled robot can be determined to a greater degree of accuracy in an indoor or outdoor environment.**

***Index Terms*—Kalman Filter, GPS, sensor fusion, Inertial Navigation System (INS).**

I. Introduction

A plethora of schemes has been proposed for determining the exact geo-location of a moving object. Determining the exact location of a robot (in the world co-ordinate system) becomes essential in multiple situations including but not limited to collaborative control techniques where each robot has to have information of either the neighboring robot or the complete formation. Various sensors are placed on the robot to determine the world co-ordinates and orientation (referred to as geo-location) of the mobile vehicle.

Global positioning system (GPS) can give the earth coordinates with a greater degree of accuracy in an outdoor environment[8]. The error tolerance in the GPS data increases exponentially when the vehicle is surrounded by high rise buildings and the GPS signal is completely inaccurate in an indoor environment[5]. Inertial navigation system (INS), consisting of the gyroscope and accelerometer (and magnetometer in some cases) gives relative information regarding the movement of the robot. The sensory information from the gyroscope has a constant drift and it has to be calibrated[9]. The data from the INS has to be filtered to maintain the validity of the data. The third sensor that is generally present on a mobile robot is the encoder placed on the wheel shaft. Odometer generally suffers from wheel slipping[5]. The data from all these different sensors can be fused to generate a good estimate of the robot's location either in an indoor or in an outdoor environment. Kalman Filter has been extensively used in sensor fusion problems. Kalman filter provides a good estimate of the desired variable when the sensor measurements are severely affected by Gaussian noise[10]. In the proposed approach noisy measurements from INS and odometer are utilized by the Kalman filter to produce an accurate estimate of the exact location of the mobile robot. The GPS data is then combined with the filtered output using a weighting scheme.

Mathematical models of all the sensors i.e. odometer, INS and GPS have been developed to generate data for testing the proposed scheme. In the INS separate model is used for accelerometer and gyroscope. The robot is modeled as a differential drive mobile robot. The model is developed in Matlab/Simulink. The robot moves on an arbitrary trajectory. The position, velocity and accelerations of the robot is fed as an input to the sensor models. Consequently, sensor data is generated that is processed by the proposed scheme. The purpose of generating sensor data was to validate the proposed sensor fusion scheme.

Body axis frame of reference refers to the co-ordinate system that is aligned with the sensor body. A sensor generates signals in body frame, these signals have to be converted to inertial reference frames using rotation matrix. The non-moving reference frame is called Inertial frame. It is an earth-fixed set of axes[9].

The paper has been organized as follows: Section II contains the robot model. Modeling of INS has been explained in section III. GPS modeling is discussed in Section IV. Section V presents the odometer model, In Section VI, the proposed sensor fusion technique is discussed. Section VII presents the implementation of the proposed scheme. Finally Section VIII presents the simulation results and Section IX concludes the paper.

II. Robot Model

The robot dynamical model from [1] has been used for testing the proposed scheme. The robot model has been used to

determine the 'x' and 'y' components of robot trajectory as well as x and y components of its velocity and acceleration. We selected this model because it gives us complete information about the robot as it not only generates the x and y co-ordinates position of robot but also gives heading angle 'ψ' information. Also, we can extract the velocity and acceleration of robot in x and y co-ordinates from this model.

$$\ddot{x} = -\dot{y}\dot{\psi} + \frac{\cos(\psi)}{mr}(\tau_1 + \tau_2) \quad (1)$$

$$\ddot{y} = \dot{x}\dot{\psi} + \frac{\sin(\psi)}{mr}(\tau_1 + \tau_2) \quad (2)$$

$$\ddot{\psi} = \frac{L}{Ir}(\tau_1 - \tau_2) \quad (3)$$

where,
'x' is robots position in x- direction
'y' is robot position in y- direction
' ψ ' is the yaw or heading angle
'm' is mass of robot
'I' is moment of inertia of robot
'L' is the length of robot
'r' is the radius of wheel
' τ_1 ' and ' τ_2 ' are torques on robot wheels

In Simulink model for robot, the following values were aassigned to the above parameters: m = 80 kg,I = 2 kg.m^2. L = 0.325 m, r = 0.075 m. To supply the torque input to robot model, an input torque is implemented that calculates torque on robot using the motor revolutions per minute (rpm). For simulation purpose the motor rpm is selected to be 70. Knowing the wheel radius of robot, the velocity in m/s can be calculated as:

$$velocity = 2\pi r * \frac{rpm}{60} \quad (4)$$

The torque on robot wheels is given by:

$$\tau = Fr \quad (5)$$

or

$$\tau = (ma)r \quad (6)$$

from the mass of robot, wheel radius and acceleration the total torque at any instant of time is calculated using (6). Since our robot model has two torques, the torque per wheel are:

$$\tau_1 = \tau_2 = \frac{\tau}{2} \quad (7)$$

III. MODELING OF INS

The inertial navigation system (INS) consists of accelerometers and gyroscope.

a. Accelerometer

An accelerometer is used to sense accelerations in two as well as three dimensions. In our Simulink model, an accelerometer model is used to produce simulated acceleration signals as if the signals are sensed by real accelerometers. Acceleration in the 'x' and 'y' direction that are generated by the robot model is fed as an input to the accelerometer model where accelerometer bias and white noise is added to them. Since, we are modeling a digital accelerometer, with an embedded ADC. The scale factor i.e LSB/g (parameter that is listed in accelerometer specification sheets) is used to obtain ADC data as given out by accelerometer in the form of a digital number[7]. To obtain acceleration in units of 'g' the digital number is divided by the scale factor (LSB/g). The accelerometer measures accelerations in body axes therefore, it is necessary to convert these accelerations from body axes to inertial frame axes using the rotation matrix. Therefore, a body axes to inertial frame axis converter block is implemented which converts body axes accelerations to inertial axes accelerations. The block is implemented using the following rotation matrix:

$$R_b^i = \begin{bmatrix} c(\psi)c(\theta) & c(\psi)s(\phi)s(\theta) - c(\phi)s(\psi) & s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta) \\ c(\theta)s(\psi) & c(\phi)c(\psi) + s(\phi)s(\psi)s(\theta) & c(\phi)s(\psi)s(\theta) - c(\phi)s(\psi) \\ -s(\theta) & c(\theta)s(\phi) & c(\phi)c(\theta) \end{bmatrix}$$

Where,

$R_b^{\ i}$ is the transformation from body axes to inertial axes. 'c' represents trigonometric function, 'cos'. 's' represents trigonometric function, 'sin'. ϕ, θ, ψ are roll pitch and yaw angles respectively.

b. Gyroscope

A gyroscope is used to detect orientation information i.e, the roll (p), pitch (q) and yaw (r) rates. Integrating these angular rates give the orientation angles. Since our robot is moving in X-Y plane and rotating with respect to Z-axis, therefore, we are interested only in the yaw angle ψ. In this case, the roll rate and pitch rate are zero therefore the corresponding angles are also zero. The yaw angle rate produced by the robot model enters the gyroscope block where noise such as gyroscope bias and white noise is added to create a noisy signal that is output by a gyroscope. Since, we are modeling a digital gyroscope, the gyroscope block contains embedded ADC that converts noisy angular rates into a digital number using a scale factor. To extract the angular rates, the digital number produced by the gyroscope is divided by the scale factor, which is listed in gyroscope specification sheet[7]. The heading angle rate (yaw rate) information obtained by integrating equation (3) of the robot model enters the gyroscope block where gyroscope bias and noise is added to it in order to produce a simulated gyroscope sensor output. The bias and noise information is obtained from gyroscope data sheets. We have implemented a three axis digital gyroscope which takes as input three angular rates as input namely roll, pitch and yaw rate and produces noisy angular rates outputs. The roll and pitch rates in our case is zero, because the robot is moving about the z-axis only. Using the yaw angular rate (r) from gyroscope model, we can calculate the yaw angle either by integrating yaw angle rate or by using Euler parameters. The first method which involves

integration will not give correct values for the desired angle due to noise added in the gyroscope block. Therefore, the noise integrates over time giving unexpected results. In the second approach, the Euler angle is used that calculates the yaw angle effectively. The following equations are implemented in this block.

Steps for calculating yaw using Euler angles

Step # 1 e_0, e_1 , e_2 , e_3 are four Euler parameters. Initial values of Euler parameters are calculated as[9]:

$$e_{oi} = \cos(\frac{\psi_i}{2})\cos(\frac{\theta_i}{2})\cos(\frac{\varphi_i}{2}) + \sin(\frac{\psi_i}{2})\sin(\frac{\theta_i}{2})\sin(\frac{\varphi_i}{2}) \quad (8)$$

$$e_{1i} = \cos(\frac{\psi_i}{2})\cos(\frac{\theta_i}{2})\sin(\frac{\varphi_i}{2}) - \sin(\frac{\psi_i}{2})\sin(\frac{\theta_i}{2})\cos(\frac{\varphi_i}{2}) \quad (9)$$

$$e_{2i} = \cos(\frac{\psi_i}{2})\sin(\frac{\theta_i}{2})\cos(\frac{\varphi_i}{2}) + \sin(\frac{\psi_i}{2})\cos(\frac{\theta_i}{2})\sin(\frac{\varphi_i}{2}) \quad (10)$$

$$e_{3i} = \cos(\frac{\psi_i}{2})\sin(\frac{\theta_i}{2})\sin(\frac{\varphi_i}{2}) + \sin(\frac{\psi_i}{2})\cos(\frac{\theta_i}{2})\cos(\frac{\varphi_i}{2}) \quad (11)$$

where, $\varphi_i, \theta_i, \psi_i$ are initial values of roll , pitch and yaw angles respectively. In our simulation, these initial values have been assigned the following values:

$$\phi_i = 0, \theta_i = 0, \psi_i = -\frac{\pi}{6} radians$$

Step # 2 The trajectories of Euler parameters are calculated as:

$$\dot{e}_0 = -\frac{1}{2}(e_1 p + e_2 q + e_3 r) \quad (12)$$

$$\dot{e}_1 = \frac{1}{2}(e_o p + e_2 r - e_3 q) \quad (13)$$

$$\dot{e}_2 = \frac{1}{2}(e_o q + e_3 p - e_1 r) \quad (14)$$

$$\dot{e}_3 = \frac{1}{2}(e_o r + e_1 q - e_2 p) \quad (15)$$

Step # 3 From these trajectories, the heading angle 'ψ' is calculated as:

$$\psi = \cos^{-1}\frac{[e_o^2 - e_1^2 - e_2^2 - e_3^2]}{\sqrt{(1-4(e_1 e_3 - e_o e_2)^2)}} sign(2[e_1 e_2 + e_o e_3]) \quad (16)$$

IV. MODELING OF GPS

The GPS signals are modeled using the x and y location information from the robot model as:

GPS_x/y= (x/y location information from robot model)+σ*Gaussian Noise (17)

where '**σ**' is the standard deviation.

The model of the robot given in equations (1) and (2) are integrated twice to get the 'x' and 'y' positions f the robot respectively. The simulated GPS signals for x and y position of robot can be generated by adding Gaussian noise with standard deviation '**σ**' to both x and y position trajectories of the robot.

V. ODOMETER MODEL

The odometer gives the speed with which the robot wheel is moving. The odometer measurements are modeled by adding noise to the velocity of the robot. Using the gyroscope measurements for heading angle'ψ' the distance traversed by the robot using odometer data is calculated by integrating the following two equations:

$$\dot{x} = v(t)\cos(\psi) \quad (18)$$

$$\dot{y} = v(t)\sin(\psi) \quad (19)$$

where,

$v(t)$ is simulated velocity from odometer and

ψ is the heading angle obtained from gyroscope model.

VI. SENSOR FUSION SCHEME

The block diagram of sensor fusion scheme is given in fig1. Output from INS and odometer is combined using Kalman filter to produce estimated values of 'x' and 'y' positions of the robot. The estimated signal and GPS signals are then weighted to produce the final output. The advantages of this scheme is that we can utilize the GPS signals to estimate more precise location information when GPS signals are available in addition to odometer and INS signals [8]. Further, this scheme can be used in indoor as well as outdoor applications where the GPS signals are available.

Weighting Parameter 'α':

In this scheme, the weighting parameter, 'α' is assigned a value of '1', when there are bad or no GPS signals and the value of 'α' lies in the range [0 1], when GPS signals are available.

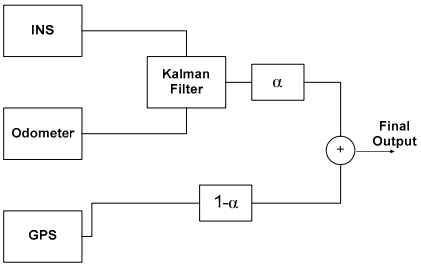

Fig. 1. Sensor Fusion Scheme for INS, odometer and GPS

Kalman filter

The Kalman filter algorithm consists of two steps: prediction and correction. Kalman filter requires 1)a mathematical model of the system whose states are to be estimated and 2) sensor measurements. Different methods to obtain the sensor measurements have been discussed earlier. The robot model is

discussed in the next section. The algorithm consists of the following steps[10]:

1. **Set the initial values of states and error covariance matrix.**

$$x\hat{o}, Po$$

2. Predicting state and error covariance matrix.

$$\hat{x}_k = A\hat{x}_{k-1} + Bu \qquad (20)$$

$$P_k^- = AP_{k-1}A^T + Q \qquad (21)$$

where,
A is the state transition matrix,
B is the control input matrix which applies the effect of each input 'u' to the state vector
Q is the process noise covariance matrix and
P is the error covariance matrix.

3. **Compute the Kalman Gain**

$$K_K = P_K^- H^T (HP_K^- H^T + R)^{-1} \qquad (22)$$

where, H is the observation matrix
R is the measurement noise covariance matrix

4. Calculate the state estimate:

$$\hat{x}_k = \hat{x}_k^- + K(z - H\hat{x}_k^-) \qquad (23)$$

5. **Calculate the error covariance:**

$$P_k = P_k^- - KHP_k^- \qquad (24)$$

VII. IMPLEMENTATION

In order to determine the exact location of the robotic platform a six state Kalman Filter is implemented.

The system state vector is:

$$X = [ax \quad ay \quad vx \quad vy \quad x \quad y]$$

where,
'ax' is the robot acceleration in x – direction.
'ay' is the robot acceleration in y-direction.
'vx' is the robot velocity in x – direction.
'vy' is the robot velocity in y – direction.
'x' is the robot position x- co-ordinate.
'y' is robot position y – co-ordinate.

The sensor measurements are:

$$Z = [ax\_acc \quad ay\_acc \quad x\_od \quad y\_od]$$

Where ax_acc and ay_acc are accelerations measured from accelerometers, and x_od and y_od are 'x' and 'y' positions determined from odometer readings. The state matrices are:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ t & 0 & 1 & 0 & 0 & 0 \\ 0 & t & 0 & 1 & 0 & 0 \\ 0.5t^2 & 0 & t & 0 & 1 & 0 \\ 0 & 0.5t^2 & 0 & t & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The observation matrix 'H' is:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The observation matrix 'H' is basically a transformation matrix that is used to map the parameters in the state vector in the measurement domain. The 'H' matrix is formulated such that pre-multiplying it with the state vector gives estimated values of those states that are also measured using sensors. Once we obtain the sensors measurements i.e. vector 'z' , the prediction error ε is calculated using the estimated state vector $\hat{x}_k^-$ and the observation matrix 'H' as :

$$\varepsilon = z - H\hat{x}_k^- \qquad (25)$$

The prediction error in equation (25) is then used to correct the state estimate using Kalman gain 'K'. Process Noise Covariance matrix:

$$Q = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}$$

Since, the number of estimated states is six, therefore the process noise covariance matrix has dimensions of 6x6.The process noise covariance matrix is assigned small arbitrary values. The measurement noise covariance matrix is:

$$R = \begin{bmatrix} \sigma^2_{ax\_acc} & 0 & 0 & 0 \\ 0 & \sigma^2_{ay\_acc} & 0 & 0 \\ 0 & 0 & \sigma^2_{x\_od} & 0 \\ 0 & 0 & 0 & \sigma^2_{y\_od} \end{bmatrix}$$

The error covariance matrix P is initialized to:

$$P_0 = Q$$

The system states are initialized to all zeros assuming no prior knowledge:

$$X_0 = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$$

For simulation, the standard deviation for GPS data is σ = 1.3. Initial weights assigned to GPS and Kalman Filtered signals are 0.5 i.e the simulation weighting parameter α=0.5. A value of 0.5 indicates that we have equal belief in the filtered data as well as the GPS signal The weighting parameter of 'α' is assigned this value to observe the effect of both the GPS signals and KF estimated result on the final output. The final output is calculated as:

Final Output = α (KF Output) + (1-α)GPS Output (26)

In order to investigate the impact of α on the estimates produced by our scheme, the values of alpha are varied and simulation results are produced for α=0.1 and α= 0.9. The first case gives more weights to GPS output whereas in the second case KF filtered output has more weightage, showing that there are little or no GPS signals are present.

VIII. SIMULATION RESULTS

The simulation results are presented in Fig2 and Fig3.The actual path of robot in both x and y directions are shown in red. The KF estimated output is shown in green. The weighted output from the proposed scheme is shown in blue. It can be seen from Fig2 and Fig3 that since the initial values of x and y were initialized to zero, assuming no *a priori* knowledge about the robot's location, the KF estimate therefore takes time to converge to acceptable values. The weighted data from GPS and KF produces a better estimate of the robot's location. The proposed scheme produces estimates of the actual x and y location with minimum error and takes less time to converge than the KF output which is evident from the two graphs in Fig1 and 2. In Fig4 and Fig5, the value of weighting parameter α is arbitrarily set to 0.1 i.e. we have confidence in our GPS signal more than the KF estimated output. This presents a condition when we have good GPS signals available (in outdoor application). In this case as well the fused output converges to acceptable values in less time as compared to the KF output. In Fig6 and Fig7, we have assigned more weights to the KF estimated output i.e. α=0.9 indicating that our GPS signal is highly inaccurate. This presents a condition when we have bad GPS signals present especially in indoor applications: Since there is very little contribution from GPS signals the weighted output follows the KF estimated output. In Fig6, we can see that since initial value of 'x' is not actually zero, therefore the KF output takes longer time to converge as compared to y-positions in Fig7.

IX. CONCLUSION

The sensor fusion scheme based on Kalman filter and weighting is implemented. The measurement data taken from odometer and accelerometers is fused to get accurate estimates of robot position in both x and y directions using Kalman Filter. The estimated output of KF and GPS is combined using a weighting scheme. Appropriate weights can be calculated based on the scenario in which the robot is operating .

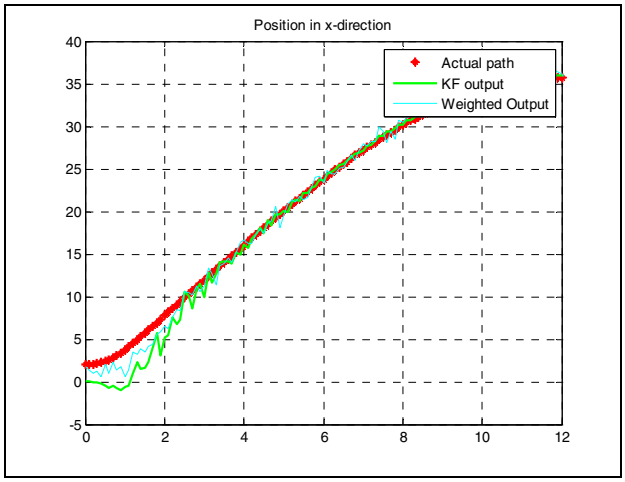

Fig. 2. Position in x-direction (α=0.5)

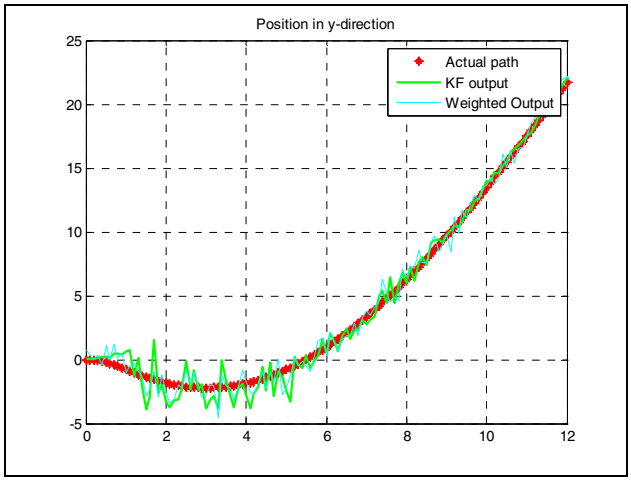

Fig. 3. Position in y-direction (α=0.5)

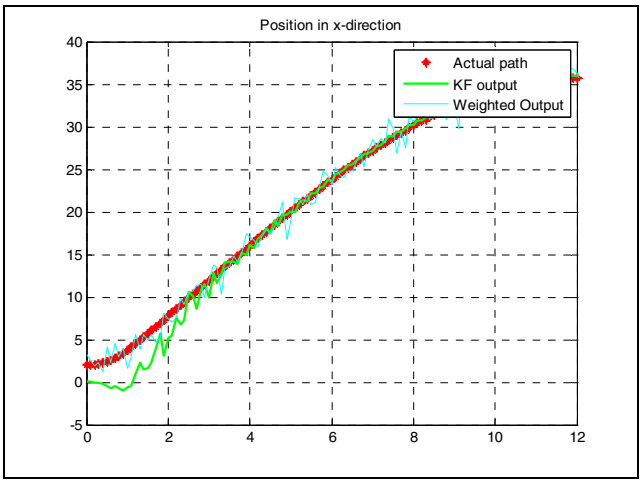

Fig. 4. Position in x-direction (α= 0.1)

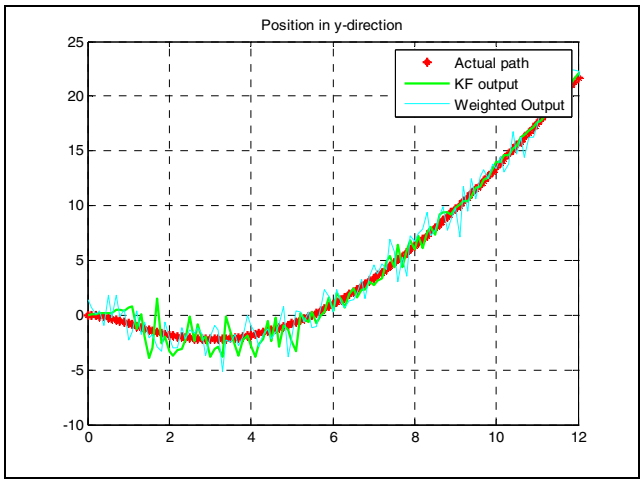

Fig. 5. Position in y-direction (α= 0.1)

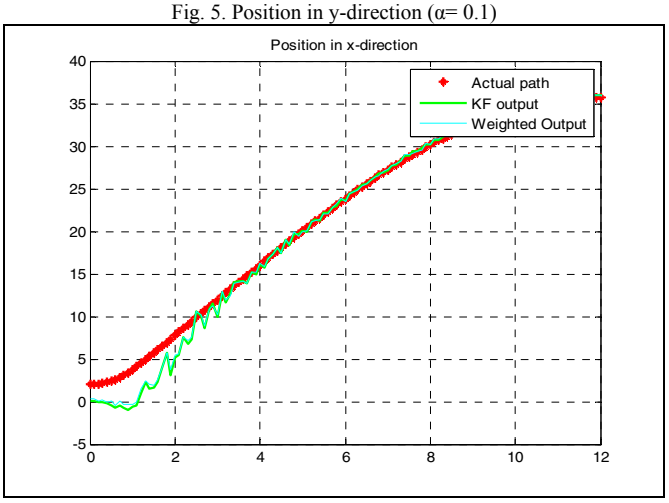

Fig. 6. Position in x-direction (α= 0.9)

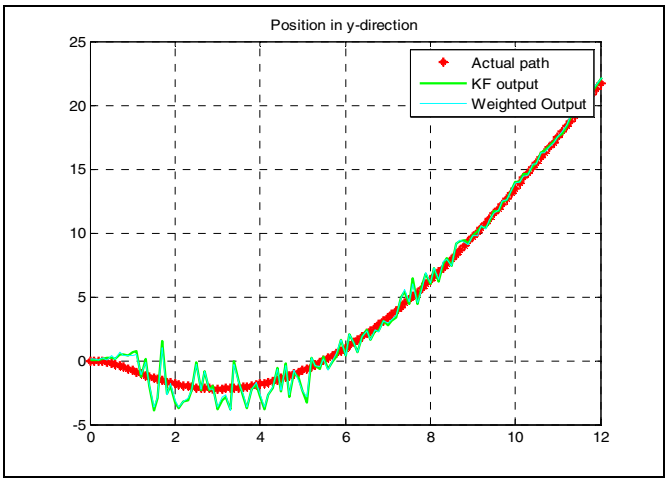

Fig. 7. Position in y- direction (α= 0.9)

It has been observed that the proposed weighted output scheme presents better results than the KF estimated output. This sensor fusion scheme can be implemented in both indoor, where the GPS signals are quite inaccurate as well as outdoor environments.

REFERENCES

[1]. M.L.Corradini, G.Orlando, "Control of Mobile Robots with uncertainties in the dynamic model: a discrete time sliding mode approach with experimental results, "Control Engineering Practice, no.10,pp.23-34, 2002.

[2] V. Bisttrovs, A.Kluga, "Combined information processing from GPS and IMU using Kalman Filtering Algorithm, "Electronics and Electrical Engineering, no.5, pp.15-20, 2009.

[3] M.S. Grewal and A.P Andrews," Kalman Filtering: Theory and Practice Using MATLAB". Wiley Interscience,January,2001.

[4] Nivedita, C. Pooja, "Object tracking in Simulink using Extended Kalman Filter", International Journal of Advanced Research in Computer and Telecommunications Engineering, Vol. 4,pp. 364-366,July 2015.

[5] K.M. Chinmaya, B.K.Mishra,J.S Jebakumar,"EKF based GPS/Odometer Data Fusion for Precise Robot Localization," IOSR Journal of Mechanical and Civil Engineering,Vol.2,pp. 70-75,April 2014.

[6] G.Welch ,G. Bishop, "An Introduction to the Kalman Filter," Technical Report :TR95-041,University of North Carolina,2006.

[7] InvenSense Inc,Product Specification, 'MPU 9150', Document Number: PS-MPU-9150A-00, Revision: 4.3,Release Date: 9/18/2013.

[8] Mayhew. D.M. "Multi-rate Sensor Fusion for GPS Navigation Using Kalman Filtering," Masters Thesis, Department of Electrical Engineering, Virginia Polytechnic Institute and State niversity,1999.

[9] K.N.Vikas, 'Integration of Inertial Navigation System and Global Positioning System Using Kalman Filter', Dept. of Aerospace Engineering, Indian Institute of Technology, Mumbai,2004.

[10] Phil Kim, Kalman Filter for Beginners with Matlab Examples", CreateSpace Independent Publishing Platform, 2011