A Project Report

On

**Auto Pilot System Using**

**PSO and RTOS**

BY

**Sidhanth Krishna 2015A7PS0088H**

**R Vivek Reddy 2015A7PS0034H**

Under the supervision of

**Prof. Chittaranjan Hota**

**SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF**

**CS F376: DESIGN ORIENTED PROJECT**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)**

**HYDERABAD CAMPUS**

**(MARCH 2018**)

**ACKNOWLEDGEMENTS**

We would like to express our sincere gratitude to our supervisor, **Prof. Chittaranjan Hota**, for providing invaluable guidance, comments and suggestions throughout the course of the project.

We are also thankful to all other and faculty members and staff members of the department for their kind cooperation and assistance and thank the institution for being extremely helpful.

Last but not the least, I would like to express my profound indebtedness towards my colleagues and my parents for providing me the moral assistance and encouragement, which helped me in successfully completing this project report.

Sidhanth Krishna – 2015A7PS0088H

R Vivek Reddy – 2015A7PS0034H

**e Redd.R**

****

**Birla Institute of Technology and Science-Pilani,**

**Hyderabad Campus**

**Certificate**

This is to certify that the project report entitled “**AUTO PILOT SYSTEM USING PSO AND RTOS”** submitted by Mr. SIDHANTH KRISHNA and R.VIVEK REDDY in partial fulfillment of the requirements of the course CS F376, Design Oriented Project Course, embodies the work done by them under my supervision and guidance.

**Date: 20/03/18 (Prof. CHITTARANJAN HOTA)**

BITS- Pilani, Hyderabad Campus

**ABSTRACT**

An autopilot is a system used to control the trajectory of any vehicle without constand hands on control by human operator being required. We belive that exploiting the prowess of **Artificial/Swarm** **Intelligence** can enhance the performance of an autopilot and to achieve this we used a genetic algorithm namely **Particle Swarm Optimisation (PSO)** which provides optimized values of measurements by searching the state space using many entities a.k.a swarm of particles and this has been of keen interest in the recent times. Our quest in this project is all about fusing the above optimization with the ever green **Kalman filter** in an effective manner to get an optimized estimate of the measured state.

The report follows a detailed explaination of how PSO can be used to optimize a real time trajectory of a vehicle using an appropriate heuristic function. Also on the other hand is a brief about how Kalman filter provides accurate state estimates of position and velocity using **GPS** and **MEMS** sensors, otherwise inaccurate, which is needed by the PSO to get the next best state.

Of the few modes of transport the ones heavily dependent on autopilots are autonoumous aerial vehicles, aircrafts, multirotors, autonomous cars etc. To illustrate the above hypothesis we plan on using one of the forms of an aerial vehicle such as an aircraft or a quad rotor. This is done so keeping in mind the **Brachistone problem** which requires a constant force field in one of the directions, in our case is the gravity. The solution of the trajectory from the brachistone’s is a **cycloid** which is rather a simple curve when compared to the rest of the dynamics models and thus the simplicity. Our objective is to achieve this trajectory that takes the least amount of time and this happens during the landing phase of an aircraft. Thus we deduce that cycloid is the necessary heuristic function to automate landing.

To compensate for the dynamics of the aerial vehicle, there should be corrections made for the stability from time to time. The moment corrections are taken care by using a combination of **Madgwick filter** and the **PID** controller. The expected loop refresh rate is calculated to be **125Hz** i.e the microcontroller makes 125 corrections every second and time of flight is approximated to be about **5 minutes**.

**Keywords:** PSO, Kalman Filter, Madgwick Filter, PID, Brachistone solution.

**CONTENTS**

Title page…………………….……………………………………………1

Acknowledgements……………………………………………………….2

Certificate…………………….……………………………………….......3

Abstract……………………………………………………………….......4

1. Particle Swarm Optimization..………………………….………….......6

2. Kalman Filter………………………………………………………......9

3. Madgwick Filter and PID controller…………………………………..15

4. Hybridization of PSO and Kalman Filter……………………………..19

Conclusion…………………………………………………………….....23

References…………………………………………………………….....24

**PARTICLE SWARM OPTIMIZATION**

In computer science, **particle swarm optimization** (**PSO**) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position, but is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

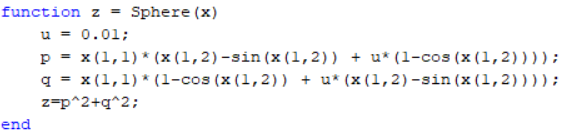
**The Algorithm:**

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered.

The objective function taken here is that of a cycloid (more about it in code) and we move a single particle along this trajectory, labelling it is an ideal particle, and thus this ideal particle determines the value of global best of the swarm and helps the non ideal particle (UAV) converge to get the characteristics of an ideal particle.

Mathematical equation that governs the fitness function is as follows:

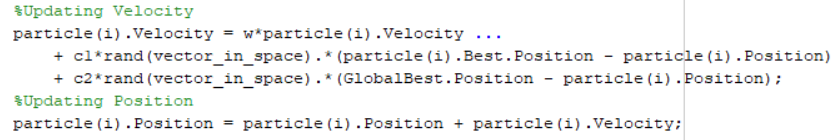
**Code:**

****

In PSO, the basic fundamental step is to manipulate the relative position of a particles of the swarm in such a way that the fitness value reduces. This is achieved by utilizing the data of a particles current position and velocity to get new position and velocity.

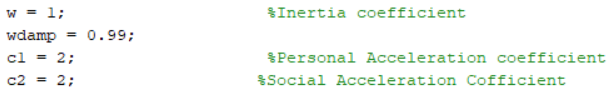
Mathematical equation that governs the swarm is as follows:

**Code:**

****

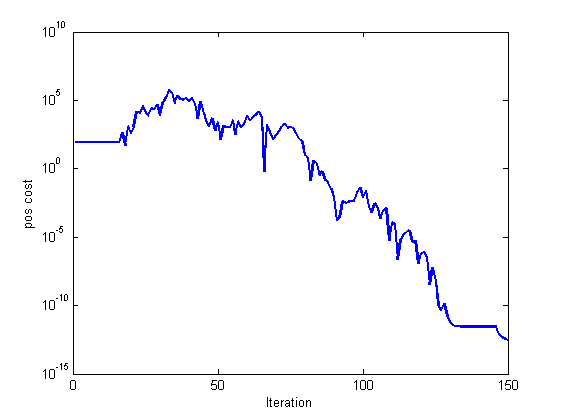
The unknown variables w, C1, C2 are parameters of PSO and they have a huge impact on the optimization. (r1, r2 signifies normal probability distrbution of particles)

**Code:**

****

**Results:**

Using the above equations we get the following simulations which proposes that the problem is convergent (as cost decreases with every iteration) and thus the given objective function can be optimized. Nevertheless, the parameters of this algorithm still needs to be fine tuned to get better results.

****

**KALMAN FILTER**

The Kalman filter is a set of mathematical equations that provides an efficient computational (iterative) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.

The Kalman filter is a very powerful tool when it comes to controlling noisy systems.

The basic idea of a Kalman filter: Noisy data in ⇒ Hopefully less noisy data out

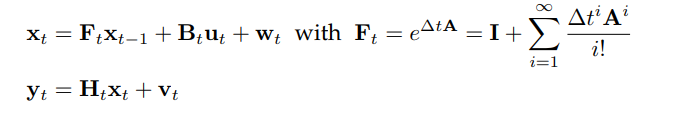
**Noise:**

In a very general sense, “noise” is an unwanted contribution to a measured signal, and there are studies on various kinds of noise related to a defined context (acoustic noise, electronic noise, environmental noise, and so forth). We are especially interested in image noise or video noise. Noise is here typically a high-frequency random perturbation of measured pixel values, caused by electronic noise of participating sensors (such as camera or scanner), or by transmission or digitization processes.

**Kalman Filter Equations:**

Vectors x0, w1, . . . , wt, v1, . . . , vt are all assumed to be mutually independent.

The defining equations of a Kalman filter are as follows:



Note that there is often an i0 > 0 such that Ai equals a matrix having zero in all of its components, for all i ≥ i0, thus defining a finite sum only for Ft.

This model is used for deriving the standard Kalman filter. This model represents the linear system with respect to time:



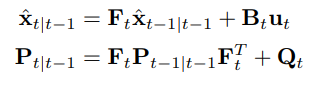
**Standard Predict-Update Equations:**

With x̂t|t we denote the estimate of state xt at time t.

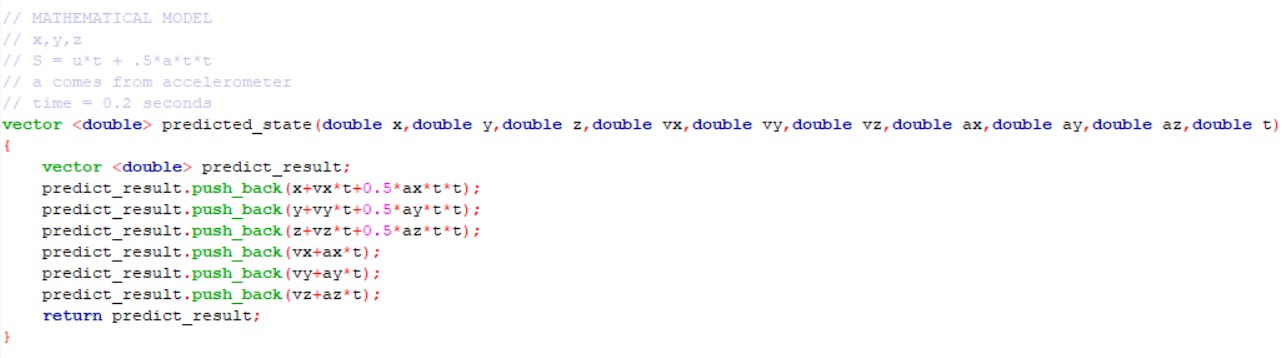
Let Pt|t be the variance matrix of the error xt – x̂t|t . The goal is to minimize Pt|t.

**Predict Phase of the Filter:**

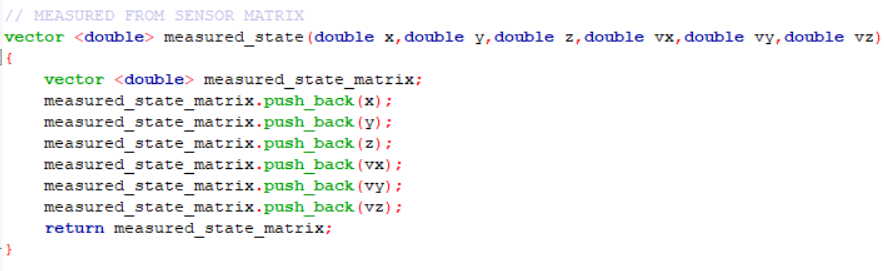
In this first phase of a standard Kalman filter, we calculate the predicted state and the predicted variance matrix as follows (using state transition matrix Ft, control matrix Bt, and process noise variance matrix Qt, as given in the model):



**CODE - 1:**

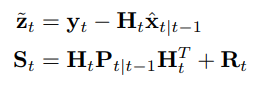
****

**CODE - 2:**

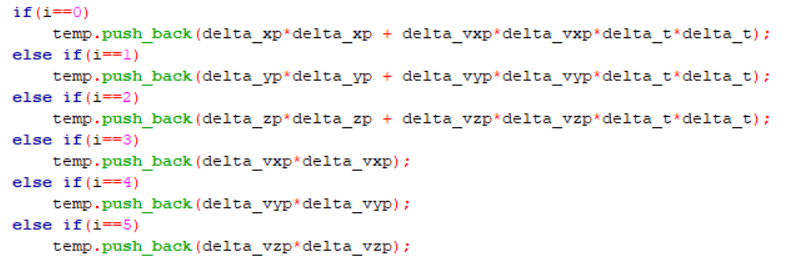
****

**Update Phase of the Filter:**

In the second phase of a standard Kalman filter, we calculate the measurement residual vector z̃t and the residual variance matrix St as follows (using observation matrix Ht and observation noise variance Rt, as given in the model):



**CODE – 1:**

****

The updated state estimation vector (i.e., the solution for time t) is calculated (in the innovation step) by a filter



**Optimal Kalman Gain:**

The standard Kalman Filter is defined by the use of the following matrix Kt known as the optimal Kalman gain:



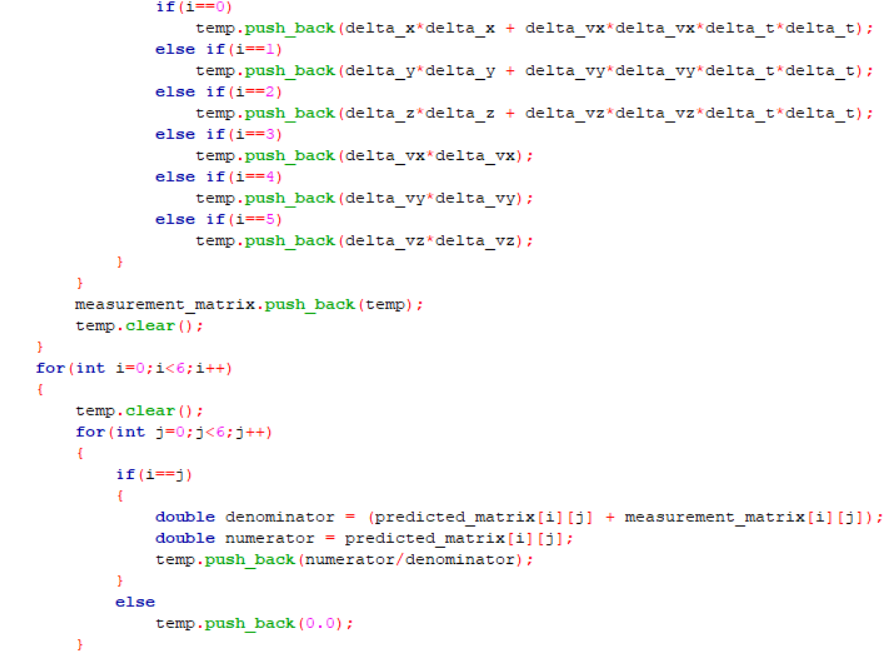
The use of the optimal Kalman gain in Equation (1) minimizes the mean square error E[ (xt – x̂t|t)2 ], which is equivalent to minimizing the trace (= sum of elements on the main diagonal) of Pt|t .

The updated estimate variance matrix

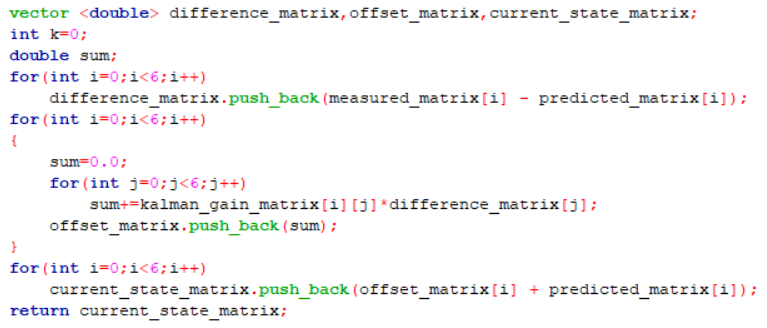


is required for the predict phase at time t + 1. This variance matrix needs to be initialized at the begin of the process. (More about it in the code)

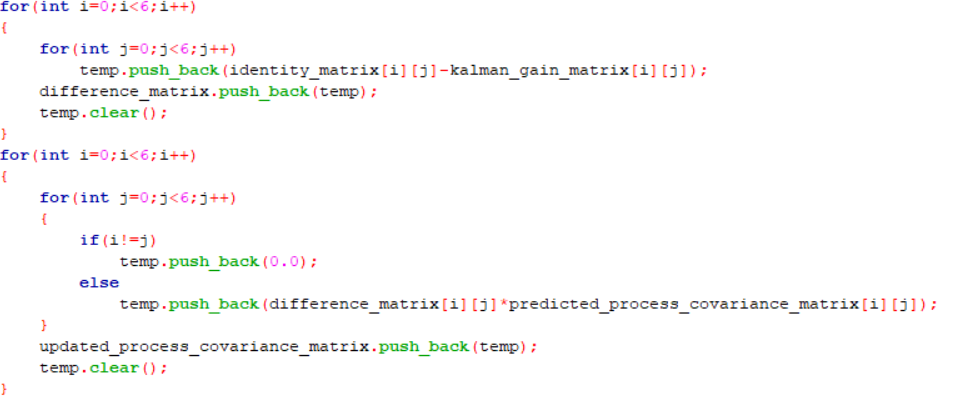
**CODE -1:**

****

**CODE - 2:**

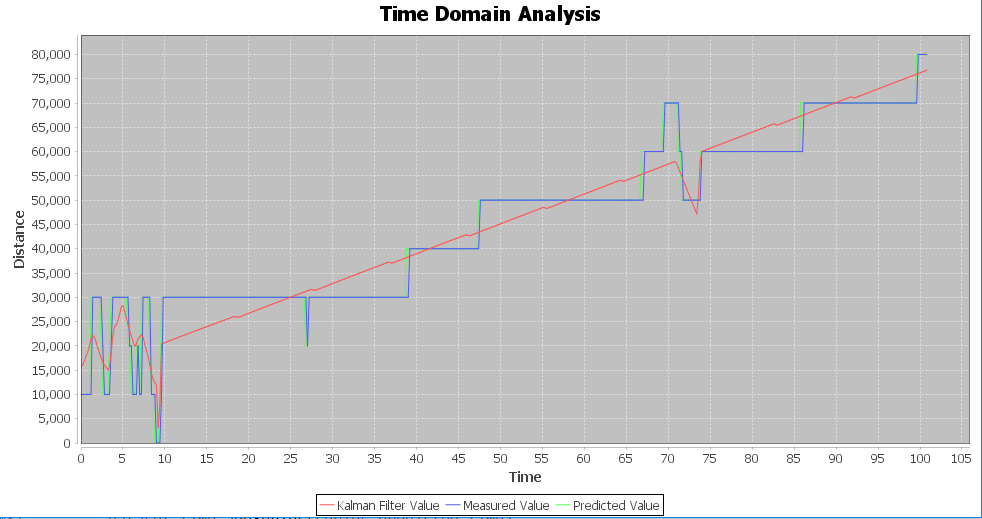
****

**CODE - 3:**

****

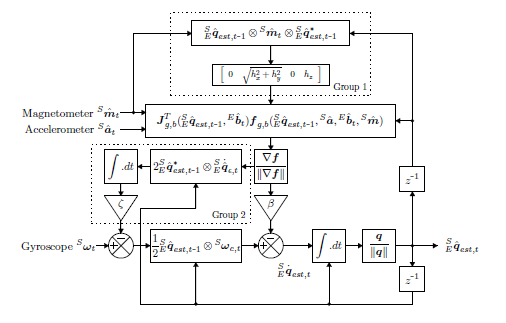
**RESULTS:**

Using the above equations we get the following simulations which proposes that the filtering operation is valid (the output is a linear combination of inputs) and stays well within the bounded range of inputs. Nevertheless, the noise matrices of this algorithm has to be included to get better results.

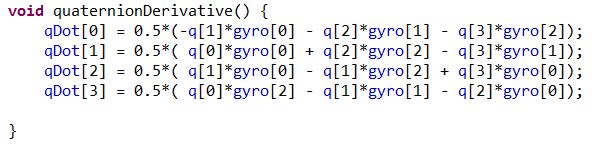
****

Madgwick filter and PID controller

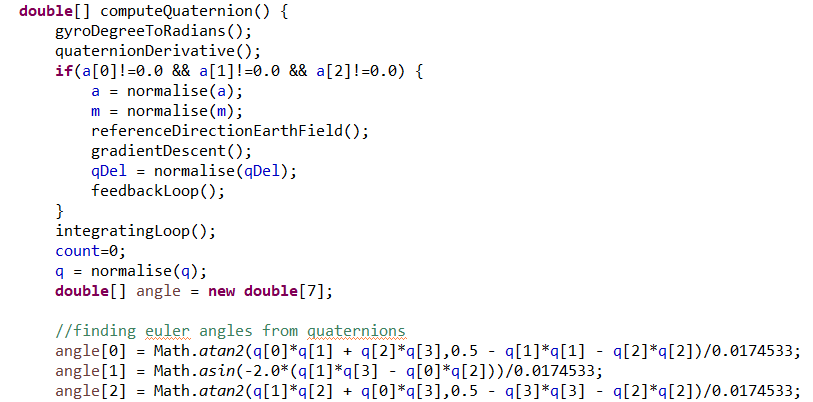
The accurate measurement of orientation plays a critical role in a range of fields. The Kalman filter has become the accepted basis for the majority of orientaion filter algorithms, however they have a number of disadvantages. They can be complicated to implement and the linear regression iterations, which are fundamental to it, demand sampling rates far exceeding subject bandwidth and thus increases computational load on the microcontroller. A novel orientaion filter known as the Madgwick filter exploits the power of a quaternion algebra to provide with accurate measurements of orientation at a much lower computational load and sampling rates. More of the above information can be found in the complete Madgwick internal report.



**CODE – 1:**

****

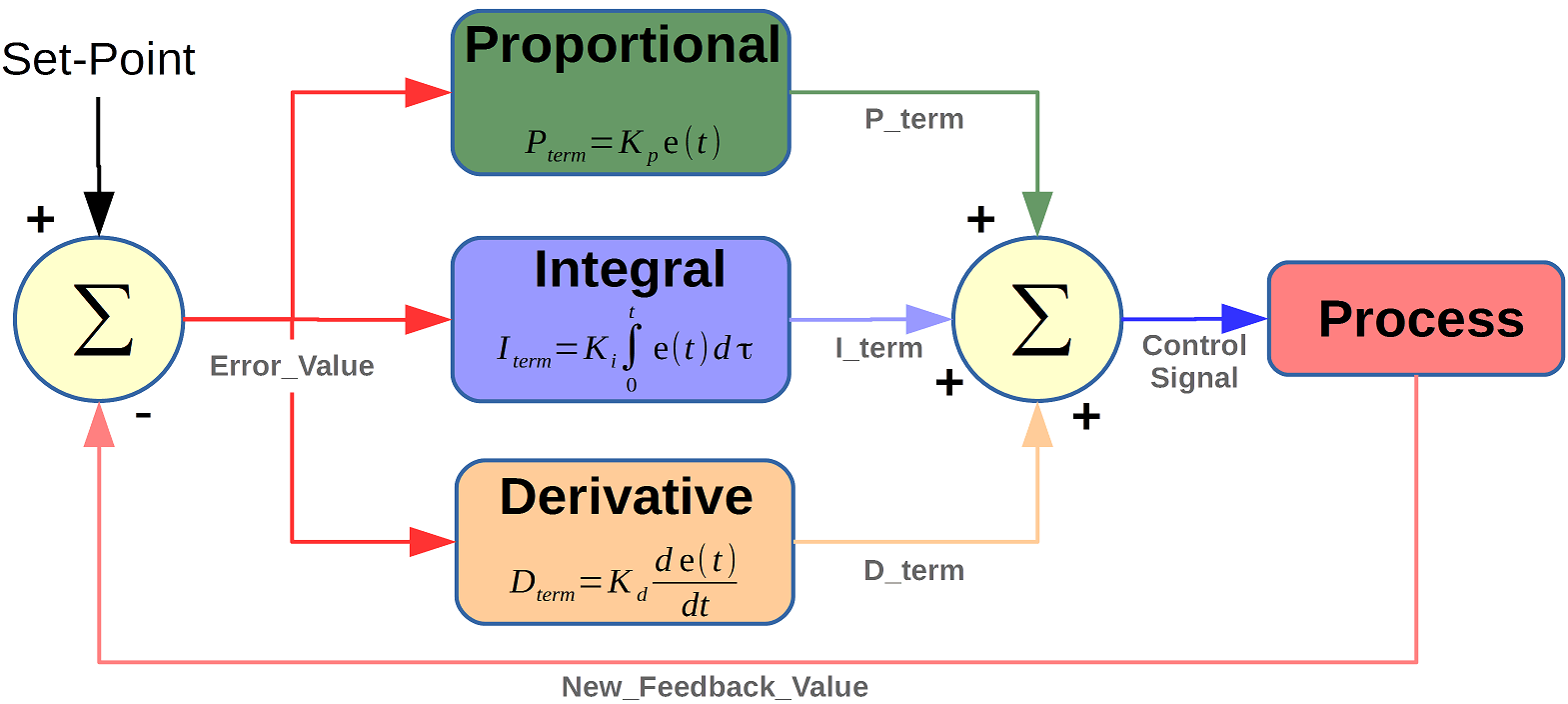
**CODE – 2:**

****

**PID controller:**

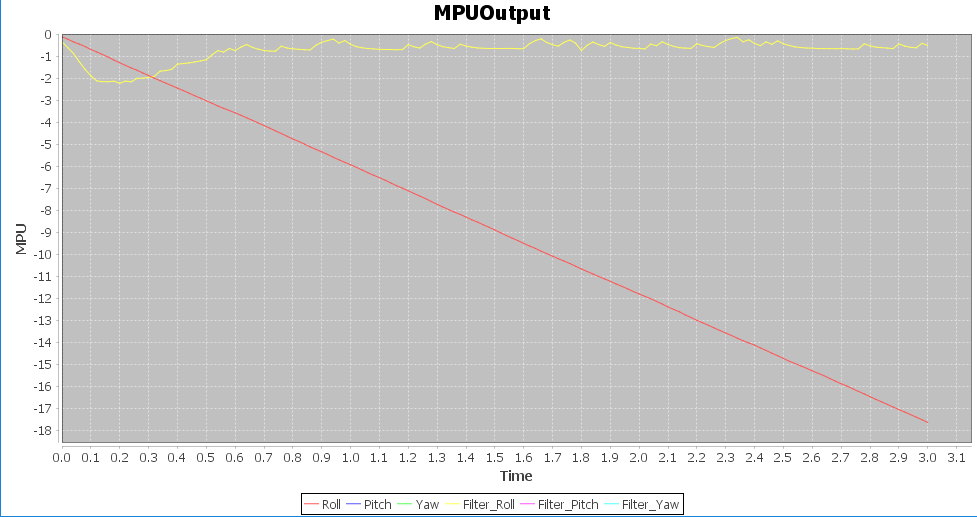
A PID controller continuously calculates an **error value** {\displaystyle e(t)}as the difference between a desired setpoint and a measured process variable and applies a correction based on proportional, integral, and derivative terms which give the controller its name.

In practical terms it automatically applies accurate and responsive correction to a control function. An everyday example is the cruise control on a road vehicle; where external influences such as gradients would cause speed changes, and the driver has the ability to alter the desired set speed. The PID algorithm restores the actual speed to the desired speed in the optimum way, without delay or overshoot, by controlling the power output of the vehicle's engine.

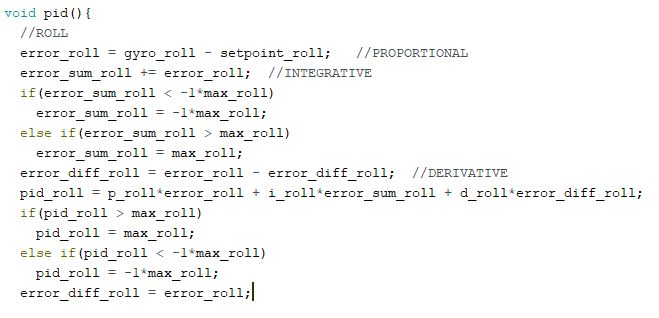


**RESULTS:**

Using the above stated filter we get the following simulations which proposes that the filtering operation is valid (the output is a linear combination of inputs from accelerometer, magnetometer and gyroscope) and stays intact even when the gyroscopic error is integrated..

****

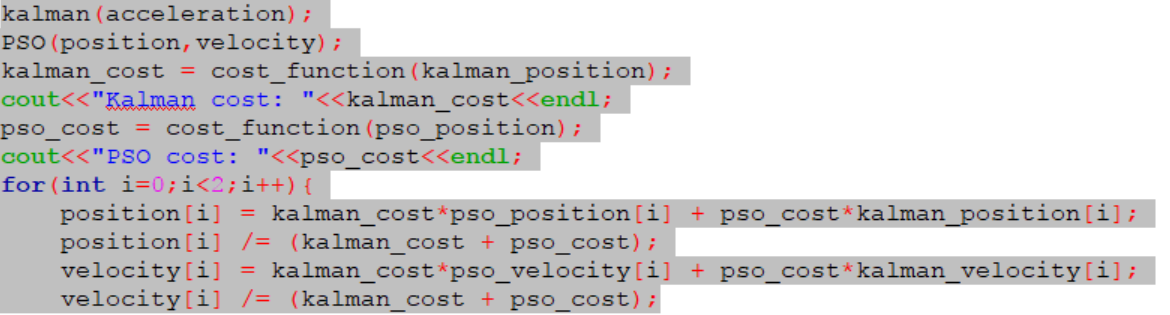
**CODE – 1:**



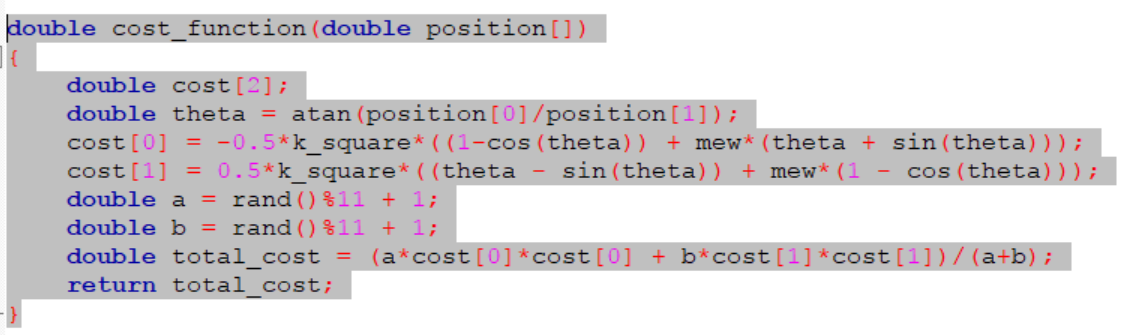
Hybridization of PSO and Kalmanfilter

The values of position and velocities computed after applying PSO and Kalman Filter are sent into an apt error function which governs the required trajectory of the particle. Let ***a*** be the weight given out by PSO and ***b*** be the value of weight given out by Kalman Filter. The new optimised estimate is hypothesized as a linear combination of the given positions with their corresponding values of inverse of their weights.

**CODE-1:**



**CODE-2:**

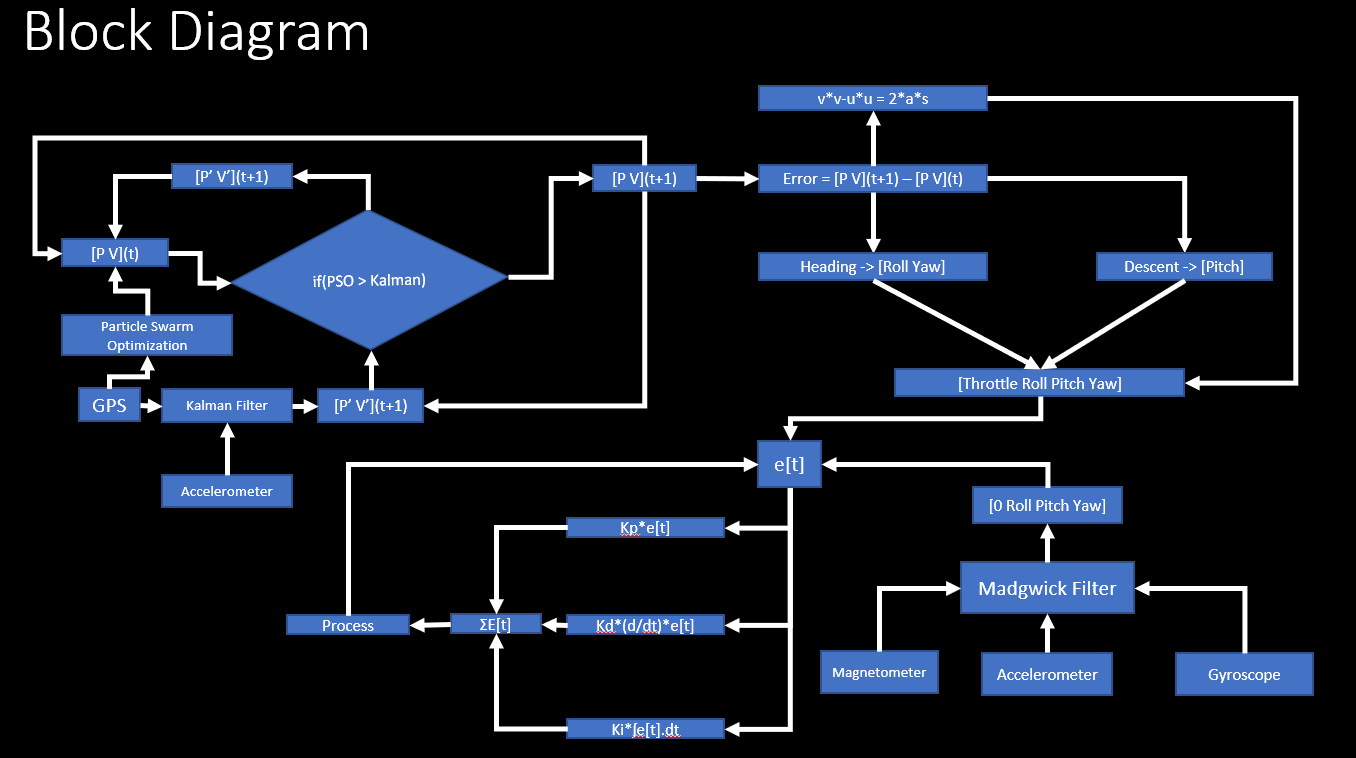


We validate the following algorithm by fetching values of mainly 3 different errors. They are as follows:

For the hypothesis to be true, the value of has to be lesser compared to and .

**SIMULATIONS:**

As the results suggest, the hybridization gave out better results than the individual parents. This hybridization consists traits of both the algorithms and gives out lesser error values. Thus we have obtained the optimized estimate of position.



CONCLUSIONS

**Work Accomplished:**

* PSO implemented on MATLAB gave positive results when plugged in with the cycloid error function.
* Kalman Filter accounted for Gaussian noise in positions and made more effective estimates.
* Used Madgwick Filter at 90 Hz on Arduino Mega. Barely passed to give satisfactory results due to poor 16MHz atmega processor.
* Hybridization of PSO and Kalman Filter was then simulated to achieve desired results.

REFERENCES

* **PSO by Mostapha Kalami Heris**  
  http://yarpiz.com/440/ytea101-particle-swarm-optimization-pso-in-matlab-video-tutorial
* **Kalman Filter by Michel van Biezen**  
  http://web.mit.edu/kirtley/kirtley/binlustuff/literature/control/Kalman%20filter.pdf
* **Madgwick Internal Report**  
  https://www.samba.org/tridge/UAV/madgwick\_internal\_report.pdf
* **PID Controller by Joop Brokking**  
  http://www.brokking.net/ymfc-al\_main.html
* **Telemetry by Farrel Farahbood**  
  https://www.youtube.com/channel/UC\_vO52hFzjMd2bPQA5AaHhg
* **Internet**