IOP Conf. Series: Materials Science and Engineering **142** (2016) 012108 doi:10.1088/1757-899X/142/1/012108

Justification of Filter Selection for Robot Balancing in Conditions of Limited Computational Resources

M V Momot[1], E V Politsinskaia[1], A V Sushko[1], I A Semerenko[1]

[1]Yurga Institute of Technology (Branch) of National Research Tomsk Polytechnic University
652055, Kemerovo Region, Yurga, 26 Leningradskaya St., Tel. (38451) 7-77-62

e-mail:mikl_1239@mail.ru

Abstract. The paper considers the problem of mathematical filter selection, used for balancing of wheeled robot in conditions of limited computational resources. The solution based on complementary filter is proposed.

I. Introduction

Right selection of compensatory mathematical filters to control the motion of autonomous remotely piloted vehicles is a relevant objective, particularly in conditions of significant lack of computational resources. The right choice of such filters solves the energy efficiency problem, in terms of robot motion and with regard to expenses for analysis and calculation. Using low-quality filters will lead to excessive robot oscillations and thus additional energy consumption conditioned by that. Filters selection implying large number of calculations leads to significant energy consumption for calculations. It is essential to find the best option, considering all the important features and factors most effectively [1,2].

II. Subject and methods of researh

To conduct experiment a model of balance robot was developed, that was equipped with stabilization system based on electronic gyroscope and accelerometer.

Electronic gyroscope differs significantly from the classical one; it does not show the angle of deviation from the given direction. Electronic gyroscope is an instrument registering and giving back the current angular velocity. Knowing the instantaneous angular velocity one can calculate approximately the rotation angle. The formula in this case looks as follows:

$$Ang_2 = Ang_1 + \Delta t \times V_{ang}, \quad (1)$$

where Ang_1 - initial angle, calculated before, Δt - short period of time, where one can consider that the velocity has not been changed, V_{ang} - value of angular velocity obtained from electronic gyroscope during Δt, Ang_2 - calculated angle, that object turned around the studied axis since the beginning of motion. For Fig.1 where axes X and Y of the gyroscope are directed parallel to the surface of motion, turning of robot is performed around the axis Z and therefore instrument readings of this axis should

be studied. In practice, it all depends on how the instrument is installed; however, for MPU-6050, in case it is placed parallel to the robot motion, it will look exactly as follows.

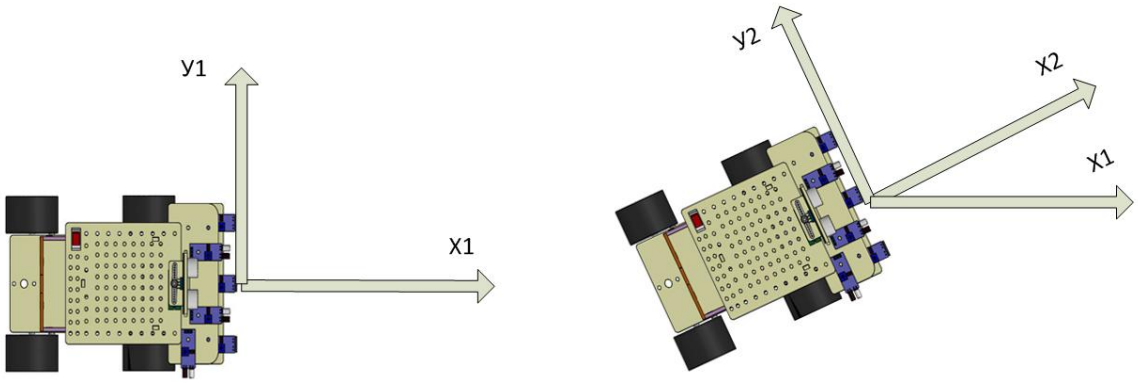

Figure 1: Demonstration of gyroscope axes and motion vector

Connection diagram of MPU-6050 to Arduino robot is given in Fig. 2.

Figure 3 demonstrates readings where the robot laid on its side, Fig.4 shows readings of the robot set parallel to the floor.

The gyroscope readings were not zero-point, even though the robot was motionless. However, they range around one value. This error can possibly be eliminated if the robot it is kept motionless for a while before it starts to move. The data is recorded at that time, the average value is calculated and then this value is subtracted from the readings obtained from the instrument. It could possibly be temperature drift, as these values are not always close to each other.

When the robot lay (Fig.3), the largest values of acceleration were observed on axis Y (>16000), while after robot turning into the normal position, the largest values were observed at the axis Z (Fig.4).

It should be taken to mind that electronic gyroscope-accelerometer MPU-6050 operates with relative values. It is produced by the company InvenSense. According to the documents, as standard, gyroscope has sensitivity of 250°/sec, to transfer gyroscope values to the scale °/sec, they are required to be divided by 131.

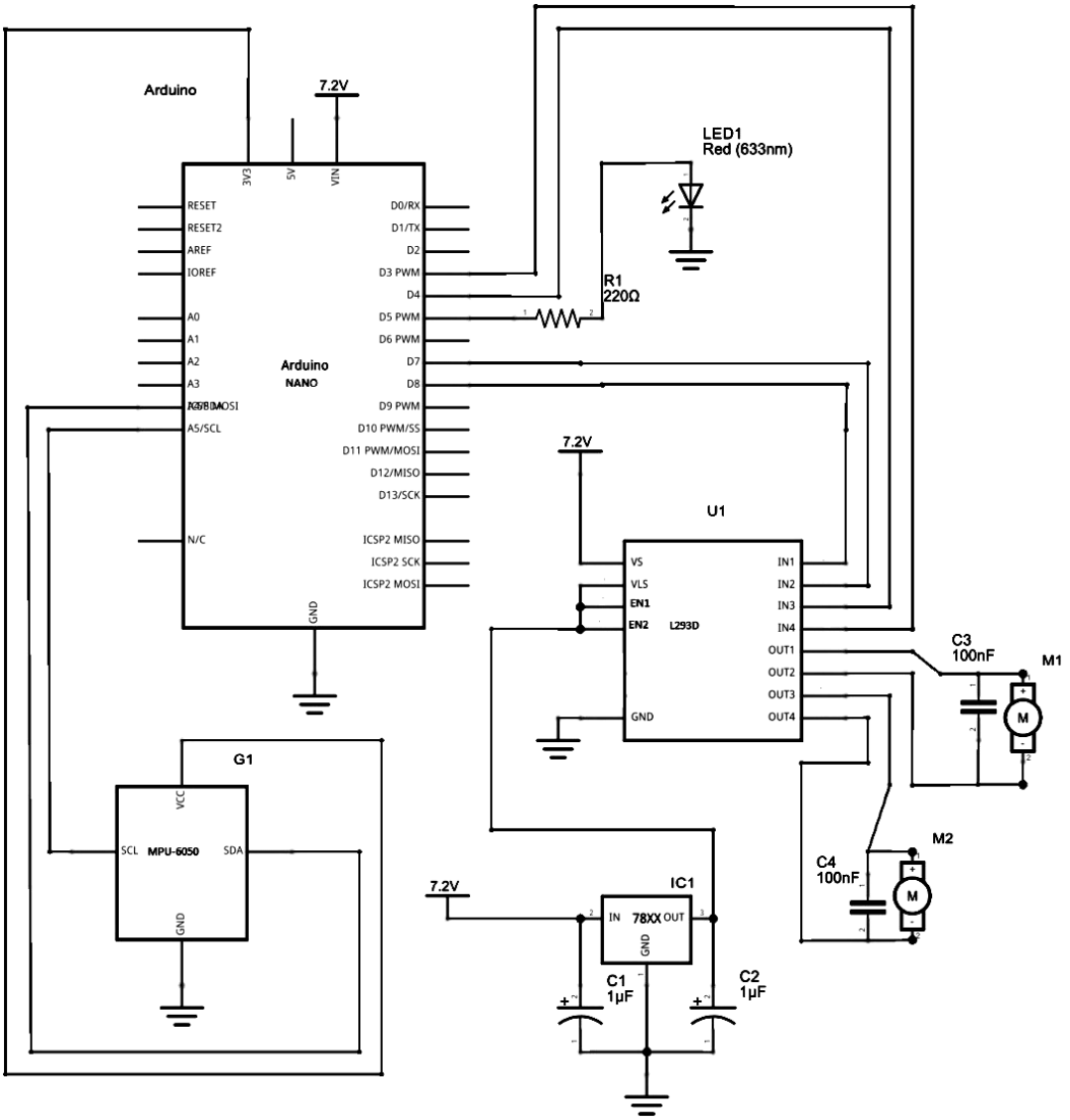

Figure 2: Connection diagram of MPU-6050 to Arduino robot

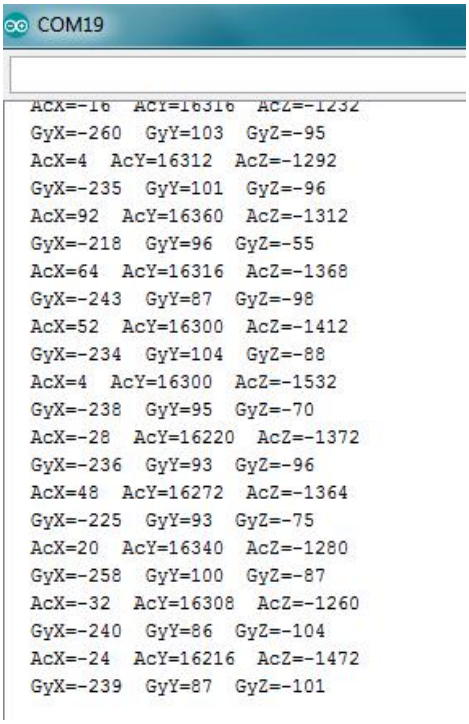

Figure 3: Readings of the robot instruments

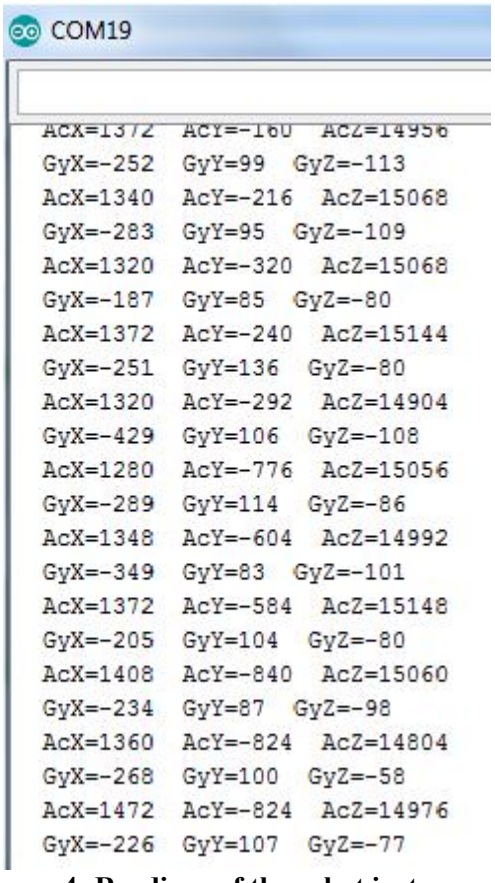

Figure 4: Readings of the robot instruments

Gyro Full Scale Range	Gyro Sensitivity	Gyro Rate Noise	Accel Full Scale Range	Magnetometer Full Scale Range	Accel Sensitivity	Digital Output	Logic Supply Voltage	Operating Voltage Supply
(°/sec)	(LSB/°/sec)	(dps/√Hz)	(g)		(LSB/g)		(V)	(V +/-5%)
±250	131	0.005	±2	N/A	16384	I²C	1.8V±5% or VDD	2.375V–3.46V
±500	65.5	0.005	±4		8192			
±1000	32.8	0.005	±8		4096			
±2000	16.4	0.005	±16		2048			

Figure 5: Sensitivity of MPU-6050

Sensitivity of accelerometer normally is 2g (g – gravitational acceleration), 16384 is 1g=9.80665 m/s². This information is quite enough for work with the instrument [2].

For this robot 2 motors with reducing gear boxes from the previous projects were used, along with circuit board ArduinoNANO, as a motor driver microcircuit L293D was used. Microcircuit L293D was soldered on a tin base, using GND contacts for that purpose. The contacts at the same time performed the role of heat removal. L293D received logical power supply from a separate stabilizer on microcircuit L7805CV. Robot obtained information red light-emitting diode LED1 to inform about the availability. Ceramic capacitors were soldered on motors to connecting terminals. Without them the robot operation was unstable, and electromagnetic interfering signals took place. The robot body was modeled using engineering software and then assembled.

Figure 7 illustrates the robot participating in testings.

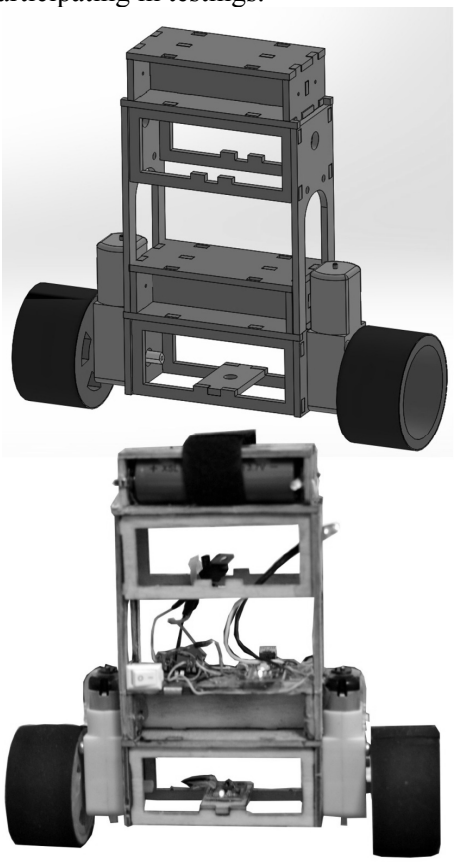

Figure 6: Robot, participating in the experiment

III. Survey results analysis

The first program (Fig.7) that was used in the present study had to analyze the angular velocity around the axis X and aimed to reduce it up to zero. In practice it looks as follows: when the robot starts to fall, it moves in that direction and compensates the bending reaching the upper part; when the robot stands still, its engines are turned off. Considering that the time intervals between polling of gyroscope are the same, time component has been reduced. The robot operates the data received directly from MPU-6050.

Zero drift. While zero angular velocity gyroscope showed non-zero velocity values around the axis X. It was compensated by introduction of the special compensator X aimed at reduction of zero drift. The robot should be set properly and then hold fix, bringing its angular velocity to zero. It is quite hard to realize in practice, however the robot response in general was correct. At the beginning of falling it was trying to get back to the initial state. However, full trim balance was not achieved.

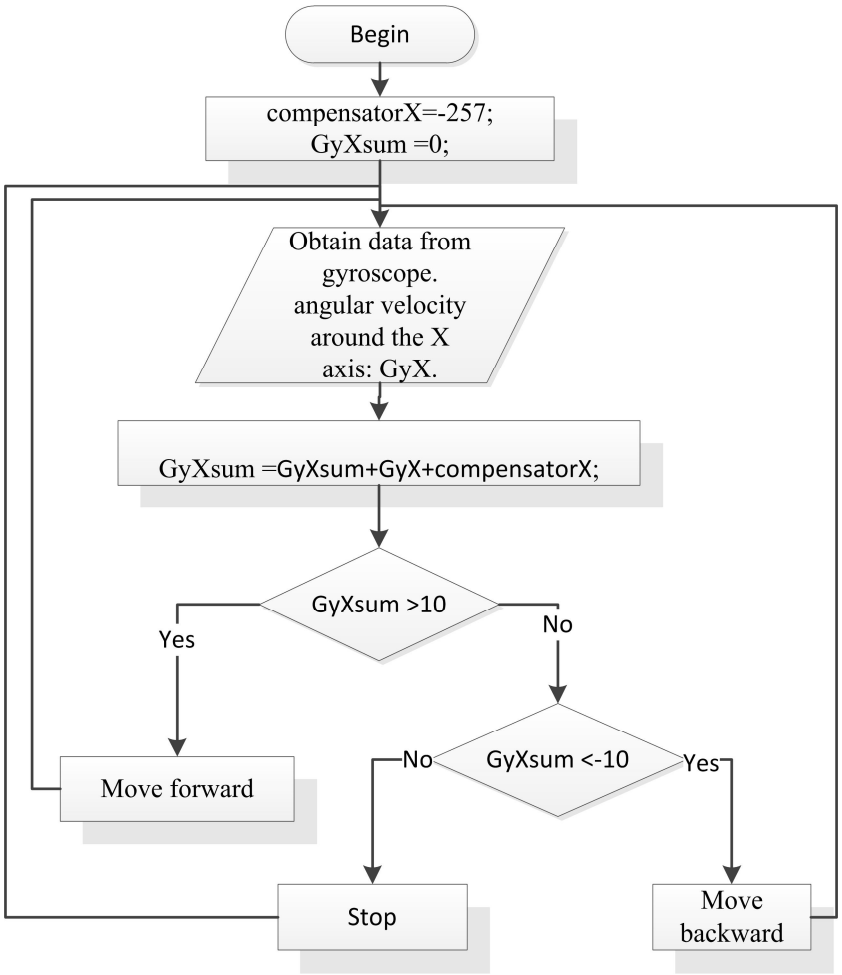

Figure 7: Simplified algorithm

We had to turn to literature and apply mathematics. Kalman filter is a recursive filter evaluating the current state of dynamic system using the data with interfering signals. Kalman filter levels jump signals successfully, transforming the values of system condition to the predictable values. The filter developer is American scientist Rudolf Kalman. While calculating the current values filter considers the previous ones and generates the current value taking them into account.

The algorithm is divided into two stages. The first stage is forecasting, the filter extrapolates the values of state variables and their indeterminancy. During the second stage the result is confirmed based on the obtained measurements.

Developer Kristian Lauszus posted on the website his own library implementations of Kalman filter [3]. In order to trim balance the robot using Kalman filter (Fig.8) one needs to record three values from MPU-6050, in this case the Y axis looks ahead while the X axis looks to the right in regards to the robot motion. These values are: acceleration values along the axes Y and Z and angular velocity along the X axis. Considering the acceleration values along the axes Y and Z we find angle of depression according to accelerometer readings (Fig.9). After that the angular velocity along the X axis is recalculated from relative units to degrees per second [4,5,6]. Then Kalman filter is called with the input parameters: angle of depression according to accelerometer readings, angular velocity along the X axis, time increment on the given area [7]. Research has shown that the filter works well, the readings are gradual but they are behind the real values thus leading to engines response being incorrect (Fig.10).

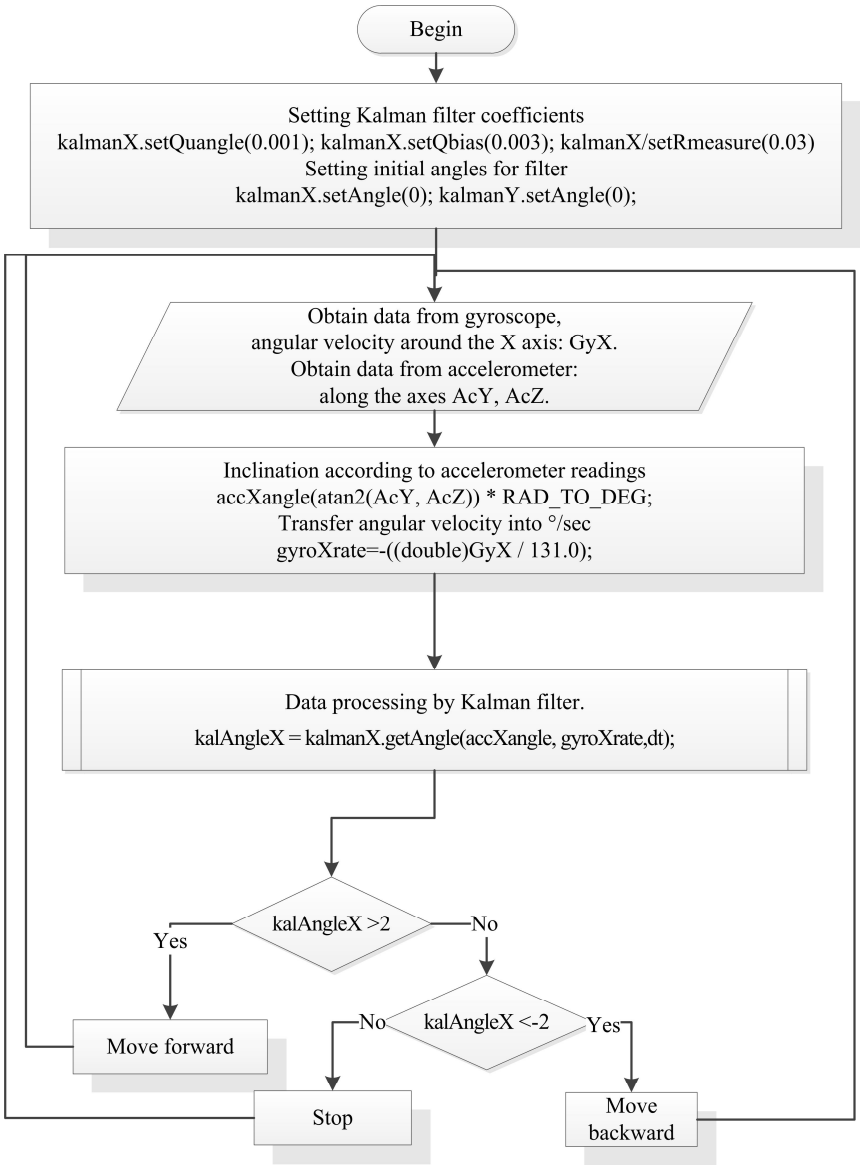

Figure 8: Algorithm with Kalman filter

The angle calculated by the program coincides visually with the preset angle (inclination angle of the robot). Further experiments demonstrated that the velocity is not enough to the filter. Filter is quite energy-consuming thus leading to problems connected with large expenses for mathematics.

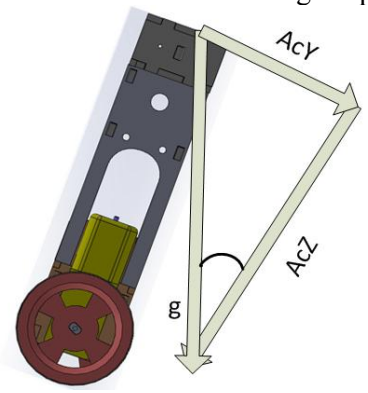

Figure 9: Demonstration of accelerometer work

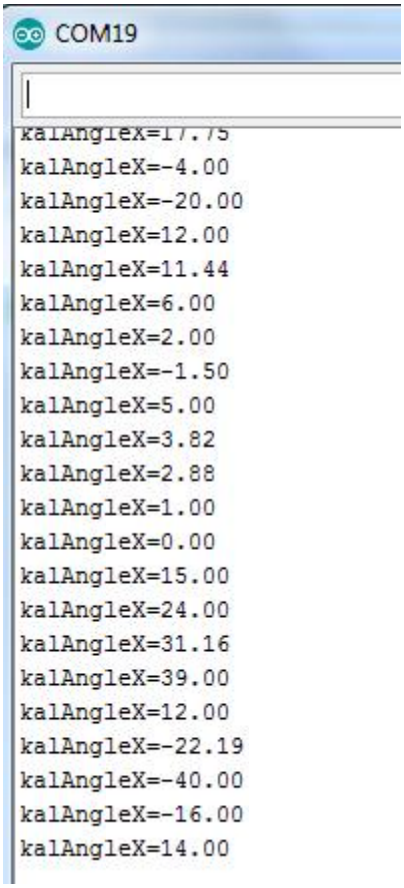

Figure 10: Instrument readings according to Kalman filter

There are other filters which are less inertial and comparatively fast, enabling to get the normal output values combining the readings from different instruments. Complementary filter refers to such filters [8,9,10].

When combining the accelerometer and gyroscope readings based on complementary filter the following formula is obtained:

$$GyXsum_1 = (1 - alfa) \times (GyXsum_0 + GyX \times \Delta t) + alfa \times AcYsum, \quad (2)$$

where $GyXsum_1$ - the calculated value of inclination angle, $GyXsum_0$ - the value of inclination angle from the previous calculation, $alfa$ - complementary filter coefficient (is selected experimentally from 0.001 to 0.05), GyX - angular velocity, Δt - iteration time, $AcYsum$- angle calculated according to accelerometer readings.

It is observed that the influence of gyroscope values on the final result is higher (0.99-0.95), however thanks to accelerometer the errors of calculations are compensated, and errors occur anyway if one uses only gyroscope. The final value is participatory sum of integrated value of gyroscope and instantaneous value of accelerometer.

Coefficient $alfa$ is accepted as 0.001; that was enough to compensate the errors of gyroscope integration and zero drift.

Thanks to using the complementary filter we have managed to stabilize the robot and it handles the objective of trim balance successfully.

During testing one peculiar feature was observed – the robot constantly moved one way. The instrument was fixed a little bit not steady in regards to the center of gravity of the robot (it was tilted). It was possible to compensate that feature by constant component added to the value of angle obtained from accelerometer. After that the robot stood still. Thus we solved the problem of motions of balancing robot, to move forward or backward one needs to change this coefficient.

The algorithm of complementary filter operation is given below (Fig.11); it is simpler for understanding and from the point of computational power consumed.

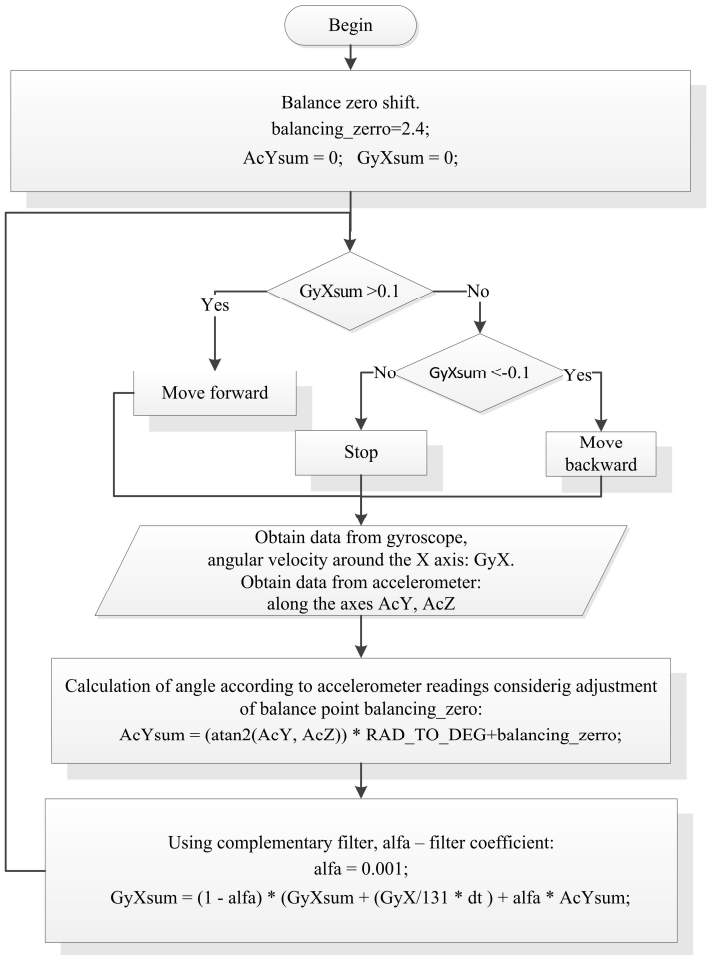

Figure 11: Algorithm using complementary filter

IV. Conclusions

Resulting from the conducted experiments it was established that taking into account the conditions of limitation of computational resources, namely when using controllers of ATMega, simple complementary mathematical filter should be used. It will allow reaching the optimum parameters of robot balancing. Filters under study behave the same way in other situations as well, e.g. while robot positioning in 3D space, defining its motion vector. Application of complementary filter enables solving the tasks of machinery positioning without using high-powered computation processors.

References

[1] Momot, Mikhail Viktorovich. Algorithm of Controlling Servo Drives by Means of Atmega 8 Microcontroller [Electronic resource] / M. V. Momot, A. Biktimirov // Applied Mechanics and Materials : Scientific Journal. — 2014. — Vol. 682 : Innovation Technology and Economics in Engineering. — [P. 596-599]
[2] Ooi, Rieh Chi (2003). Balancing a Two-Wheeled Autonomous Robot. University of Western Australia: Thesis B. Mechatronics Engineering.
[3] https://github.com/TKJElectronics/KalmanFilter
[4] Bose, I. (2000). How to Make Good Homemade PCBs? Electronics for You Magazine.Pg 37-40.
[5] R. Fierro, F. Lewis, and A. Lowe, "Hybrid control for a class of underactuated mechanical systems", IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 29, no. 6, pp. 649-4, nov 1999.
[6] N. M. A. Ghani, F. Naim, and P. Y. Tan, "Two wheels balancing robot with line following capability", World Academy of Science, Engineering and Technology, vol. 55, pp. 634-8, 2011.
[7] X. Ruan, J. Liu, H. Di, and X. Li, "Design and LQ control of a two-wheeled self-balancing robot", in Control Conference, 2008. CCC 2008. 27th Chinese, july 2008, pp. 275-279.
[8] T. Nomura, Y. Kitsuka, H. Suemistu, and T. Matsuo, "Adaptive backstepping control for a two-wheeled autonomous robot", in ICROS-SICE International Joint Conference, Aug. 2009, pp. 4687-92.
[9] K.-H. Su and Y.-Y.Chen, "Balance control for two-wheeled robot via neural-fuzzy technique", in SICE Annual Conference 2010, Proceedings of, aug. 2010, pp. 2838-2842.
[10] V. Georgitzikis and I. Akribopoulos, and O. Chatzigiannakis, "Controlling physical objects via the internet using the Arduino platform over 802.15.4 networks", IEEE Latin America Transactions, vol. 10, no. 3, pp. 1686-9, 2012.