A Realization of an FPGA Sub System for Reducing Odometric Localization Errors in Wheeled Mobile Robots

Received 14th August 2008; accepted 8th December 2008.

James Kurian, P.R. Saseendran Pillai

Abstract:

This paper introduces a simple and efficient method and its implementation in an FPGA for reducing the odometric localization errors caused by over count readings of an optical encoder based odometric system in a mobile robot due to wheel-slippage and terrain irregularities. The detection and correction is based on redundant encoder measurements. The method suggested relies on the fact that the wheel slippage or terrain irregularities cause more count readings from the encoder than what corresponds to the actual distance travelled by the vehicle. The standard quadrature technique is used to obtain four counts in each encoder period. In this work a three-wheeled mobile robot vehicle with one driving-steering wheel and two-fixed rear wheels in-axis, fitted with incremental optical encoders is considered. The CORDIC algorithm has been used for the computation of sine and cosine terms in the update equations. The results presented demonstrate the effectiveness of the technique.

Keywords: *FPGA, position estimation, robot localization, odometric correction.*

1. Introduction

Mobile robots and automated guided vehicles (AGV) operating in industrial and other environment require accurate sensing of vehicle position and attitude. In the real world environment, it is still a very challenging problem. In this era the autonomous or semi autonomous robot vehicles find applications in automated inspection systems [1], floor sweepers [2], hazardous environments [3], autonomous truck loading systems [4], agriculture tasks, delivery in establishments like manufacturing plants, office buildings, hospitals [5], etc. and providing services for the elderly [6]. In addition to this, autonomous vehicles are widely utilized in undersea exploration and military surveillance systems [7],[8]. Automated Guided Vehicles (AGVs), such as the cargo transport systems are heavily used in industrial applications. Mobile robots are also finding their way into a growing number of homes, providing security, automation [9], [10], and even entertainment. In all these applications, some type of localization system is essential. In order to navigate to their destination, the robots must have some means of estimating *where they are* and in *which direction they are heading*. The knowledge of position and attitude information is not exclusive to the realm of mobile robots. Information about the location of an inanimate object, for example a cargo pallet, can streamline inventory and enable warehouse automation. A variety of techniques have been developed and used successfully to provide the position and attitude information. However, many of these existing positioning systems have inherent limitations of their own in the workspace.

In mobile robot applications, two basic position estimation methods are employed concurrently, viz., the *absolute* and *relative* positioning [11]. *Absolute* positioning methods usually rely on the use of appropriate exteroceptive sensing techniques, like navigation beacons [12],[13], active or passive landmarks [14], map matching [15], or satellite-based navigation [16] signals. Navigation beacons and landmarks normally require costly in-stallations and maintenance, while map-matching methods are usually slower and demand more memory and computational overheads. The satellite-based navigation techniques are used only in outdoor implementations and have poor accuracy. Relative position estimation is based on proprioceptive sensing systems like odometry [17], Inertial Navigation System (INS) [18] or optical flow techniques [19], where the error growth rate of these systems are usually unacceptable. The vehicle performs self-localization by using relative positioning technique, called *dead reckoning*. For implementing a navigational sys-tem most of the mobile robots use position odometric system together with traditional inertial navigation systems employing gyros or accelerometers or both. The odometric system provides accurate and precise intermediate estimation of position during the path execution.

The Inertial Navigation System (INS) is complex and expensive and requires more information processing for extracting the required position and attitude information. The localization based on INS uses accelerometers or gyros, where the accelerometer data must be integrated twice to yield the position information, thereby making these sensors extremely sensitive to drift. A very small error in the rate information furnished by the INS, can lead to unbounded growth in the position errors with time and distance. Rate information from the gyros can be integrated to estimate the position and yields better accuracy than accelerometers. Though the odometric system is simple, inexpensive and accurate over short distances, it is prone to several sources of errors due to wheel slippage, variations in wheel radius, body deflections, surface roughness and undulations. Odometric system is reliable and reasonably accurate, on smooth flat terrain, and in the absence of wheel slippage, since a wheel revolution corresponds to linear travel distance. On paths having terrain irregularities the odometric systems are not considered to be useful, because the measured rotations of the wheels do not accurately reflect the distance travelled due to wheel slippage and motion over humps and cracks. Such odometric errors need to be corrected in practical mobile robotic applications.

This paper presents the realization of a new, simple and efficient system to reduce the errors caused by terrain irregularities like humps, cracks or other disturbances, which contribute to errors in an odometric system in a mobile robot vehicle. Redundant odometric sensors are considered in this technique. In addition to this we present an adaptive speed measurement and standard quadrature technique to improve the position and speed resolution. The system has been realized in an FPGA and the results are presented.

2. The odometric system

In this work a three-wheeled mobile robot vehicle with one driving-steering wheel and two-fixed rear wheels in-axis, fitted with incremental optical encoders is considered. The incremental encoders are the most frequently adopted position transducers for the mobile robotic applications. The low level information provided by the encoder in the form of two pulse trains namely Ch_A and Ch_B are 90° out of phase (quadrature) and depending on the direction of rotation, one of these pulses will lead or lag the other. These low level signals are passed to the control system, which computes the actual position, speed and acceleration information needed for the controller. Incremental encoders, which have been used for this purpose, are characterized by high accuracy, high resolution, high noise immunity, low maintenance and low cost and hence are generally preferred.

2.1. Position measurement

The simplified block diagram of the Encoder Pulse Processing Module (EPPM) is shown in Fig. 1. The encoder pulses Ch_A and Ch_B are fed to a quadrature decoder circuit, which decodes the direction indicator bit and generates pulses during the rise and fall fronts of both channels. A 20-bit up/down counter is implemented and the direction bit would set the counter as up or down depending upon the direction of movement of the wheel. The computational unit resets the counter after reading the position or position increment.

The position measurement involves counting the encoder pulses with the help of the quadrature decoder circuit and the direction of movement of the wheel can also be decoded from the encoder pulses. The count is incremented or decremented depending on which pulse leads the other. The present position P (or position increment) and the position error noise δP due to quantization are given by

$$P = \frac{\pi CR}{2NG};\ \delta P = \frac{\pi R}{2NG}$$

Where C is the counter value, R is the radius of the wheel in metres, N is the number of pulses per revolution of the encoder and G is the gear ratio between the encoder shaft and robot wheel coupling.

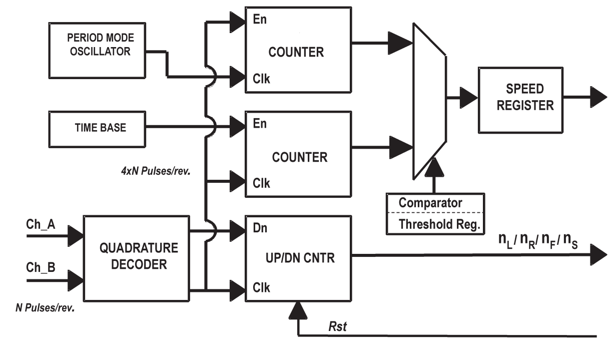

Fig. 1. Functional block diagram of the optical incremental Encoder Pulse Processing Module (EPPM), which computes the velocity information in period mode and pulse mode depending upon the current speed of the vehicle. It also provides the incremental position (distance) update of the wheel.

2.2. Speed/velocity estimation

The upper part of the encoder pulse-processing module in Fig. 1 shows the velocity measurement sections. Two major techniques for extracting the speed data from an incremental encoder are *pulse counting* and *period measurement* [21], [22], [23] of the pulses. In pulse counting mode the number of pulses C in an observation time window T is counted and the speed is approximated to the discrete incremental ratio as:

$$S = \frac{\pi CR}{2NTG}$$

The speed error due to one bit quantization

$$\delta S = \frac{\pi R}{2NTG}$$

The other method for obtaining the speed is to measure the time between two successive pulses from the encoder. In this method a stable high frequency clock (f) and gating circuit is normally used at the front end of the counter. The encoder pulses control the gating circuit, so the count value depends on the number of encoder pulses per revolution and the clock frequency. For a given system, the clock frequency and the number of encoder pulses per revolution are constant and hence the count value is inversely proportional to the speed of the vehicle and the relationship is given by

$$S = \frac{\pi f}{2NCG}$$

In pulse counting scheme the pulses from the quadrature decoder (x4) circuit is gated with the time base and fed to the counter and the final count value is latched. Instead of the direct pulses from the encoder, the quadrature-decoded pulses are used, as the pulse rate is four times higher. This reduces the switching between *period measurement* and *pulse counting* schemes.

The implementation of the period measurement scheme is very similar to pulse counting except that the counting clock is from the reference clock generator and the gating pulse is from the quadrature decoder. Hence the count value is a measure of the velocity as it counts the number of *period mode* clocks between successive pulse intervals. The system can switch between these two velocity-measuring modes by monitoring the count register value because period counting is more accurate at low speeds and pulse counting is more accurate at high speeds. The 3D plot in Fig. 2a shows the variation of error with respect to the time window T and encoder pulse per revolution N and in Fig. 2b shows the error variation with respect to encoder pulse per revolution and the period mode oscillator frequency. From these plots it is very evident that the proper switching between these two modes facilitates the accurate measurement of velocity information. If the pulse count value in the *pulse mode* counter is less than a reference value then the velocity measuring scheme switches to the period measuring mode and *vice versa*. The speed register value along with the mode bit (period/pulse) can be read to get the velocity information.

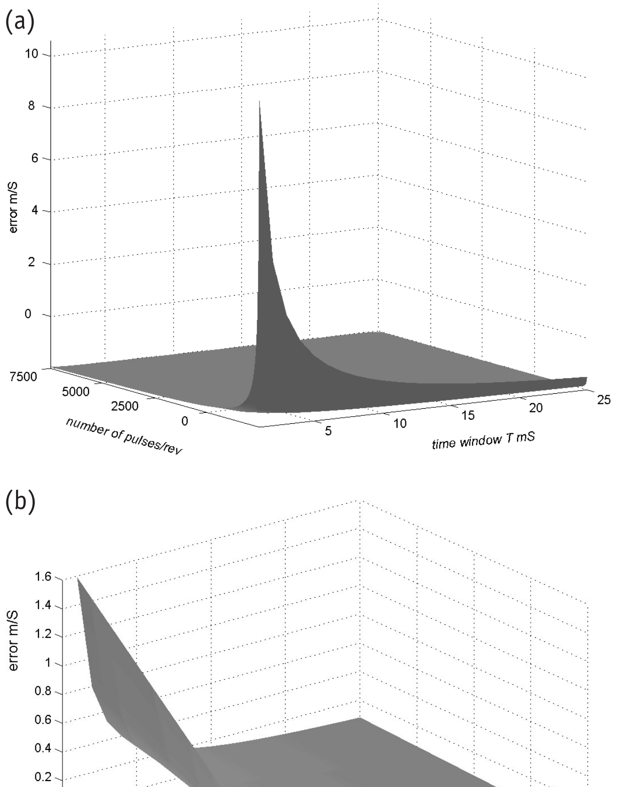

Fig. 2. 3D surface plot showing the role of (a) quantization error plotted against encoder pulse per revolution and observation time window in pulse counting mode and (b) relative error plotted against encoder pulse per revolution and oscillator clock frequency in period counting mode.

3. Position and attitude estimation

The kinematics and navigation equations for a three-wheeled mobile vehicle with one driving-steering wheel and two fixed rear wheels in-axis is considered in this study. The odometric navigational system is implemented using four optical incremental encoders. The driving steering wheel (front) is attached with geared permanent magnet DC motors with inbuilt encoder, which measures the angular increments for the measurement of the steering angle and the distance moved by the vehicle. The rear wheels are also attached with encoders to estimate the position and attitude of the vehicle. A typical pose of the mobile robot vehicle with a steering angle ϕ and an orientation θ with respect to the absolute reference frame OXY is shown in Fig. 3.

The symbols used in the equations are defined below:

$X_k\,(x, y, \theta)$ - position and attitude vector

θ -	the estimated attitude of the vehicle with respect to the fixed reference
θ_R -	the estimated attitude of the vehicle from the rear wheels encoder counts with respect to the fixed reference
θ_F -	the estimated attitude of the vehicle from the front wheels encoder counts (steering & driving) with respect to the fixed reference
ϕ -	the steering angle with respect to axis of symmetry
R -	the wheel radius of the vehicle
n_L - n_R - $n_F - n_S$ -	the encoder incremental pulse counts from the left, right, front-wheel and steering encoders respectively.
N -	the number of pulses per revolution of the encoder.
L -	the distance between the rotation axis of the front (driver) wheel and the axis of the back wheel.
D -	the distance between rear wheels

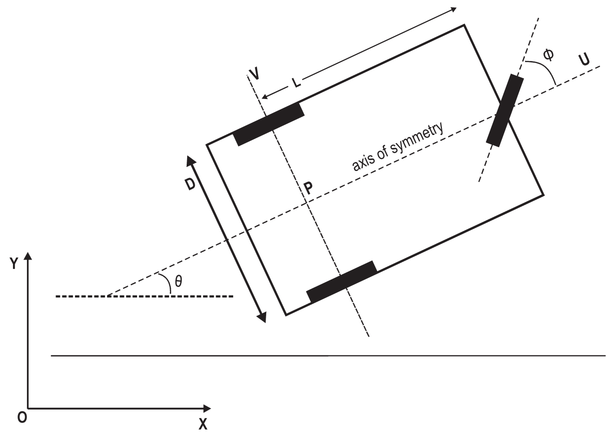

Fig. 3. The Kinematic scheme of the three wheeled mobile robot vehicle having attitude θ, which is the angle between the absolute reference frame OXY and the mobile reference frame PUV. The origin P is attached to the midpoint of the axes joining the rear wheels and the axis of symmetry of the vehicle. ϕ is the steering angle.

3.1. Position updates from encoder data

The better a vehicle's odometry, the better will be its ability to navigate and lesser will be the requirement for frequent position updates with respect to external sensors. For the computation of the position and attitude let us consider the pulse counts from the two independent optical encoders attached to the rear non-driven idler wheels of the vehicle which have less coupling with the steering and driving system and very less slippage between point of contact and the floor. The update equations for this model are as follows [20]:

$$x(k+1) = x(k) + \frac{\pi R}{N}\,(n_R(k) + n_L(k))\cos\theta(k) \qquad (1)$$

$$y(k+1) = y(k) + \frac{\pi R}{N}\,(n_R(k) + n_L(k))\sin\theta(k) \qquad (2)$$

$$\theta(k+1) = \theta(k) + \frac{2\pi R}{N}\,\frac{(n_R(k) - n_L(k))}{D} \qquad (3)$$

Normally these readings are very accurate and stable compared to the position and attitude computation with respect to the front wheel encoder data because it is less prone to slippage or skidding.

The distance moved by the wheel's point of contact could also be derived by considering the vehicle's front driving steering wheel's incremental pulse count data. The steering rotation is limited to ±40° about the axis of symmetry of the vehicle. The encoder attached to the steering system generates pulses corresponding to the steering movement and the steering angle ϕ can be computed. From these data the position and attitude of the vehicle can be estimated as follows:

$$x(k+1) = x(k) + \left(\frac{2\pi R}{N}\right) n_F(k)\cos\theta(k)\cos\phi(k) \qquad (4)$$

$$y(k+1) = y(k) + \left(\frac{2\pi R}{N}\right) n_F(k)\cos\theta(k)\sin\phi(k) \qquad (5)$$

$$\theta(k+1) = \theta(k) + \left(\frac{2\pi R}{N}\right) n_F(k)\,\frac{\sin\phi(k)}{L} \qquad (6)$$

The pulse count received from certain encoders may indicate an over count due to terrain irregularities and operating conditions. So the least value of $x(k+1)$, $y(k+1)$ and $\theta(k+1)$ estimated from the equations (1) to (6) can be used for computing the pose of the vehicle. The necessary hardware is designed and developed for the independent computation and comparison of the position and attitude values from the rear wheel and front wheel encoder data. The digital comparators manage the switching of multiplexers that selects the least values among the computed values.

3.2. The error reduction technique

Conventional systems use data from a set of encoders to compute the posture of the vehicle. By utilizing the equations (1)-(3) and reading encoder pulse counts of the rear wheels one can compute the position and attitude of the robot vehicle. Using equations (4)-(6), the posture of the vehicle can be computed by utilizing the encoder data available from the front wheel.

The technique presented here utilises both the sets of encoder values and computes the position and orientation of the vehicle. Thus redundant information for the computation of the posture of the vehicle is available. The wheel slippage, motion over humps, cracks or any other terrain disturbances cause more pulses than what corresponds to the actual distance travelled [24]. This may lead to over incremental update values of position and orientation. Independently computing the position increments and angular increments of the system and dropping the higher values eliminate this error. The new position and orientation are updated with the minimum (lower) values.

Fig. 4 shows three typical postures of the vehicle over a hump. The Fig. 4a corresponds to an error condition of over counts from rear wheel encoders that records more distance than actual. The front wheel encoders produce the data corresponding to the actual distance moved by the vehicle. Fig. 4b represent a situation where the rear wheel encoders produce over counts and over attitude errors and Fig. 4c shows the front wheel encoder pose over a hump that may cause over count from the encoder.

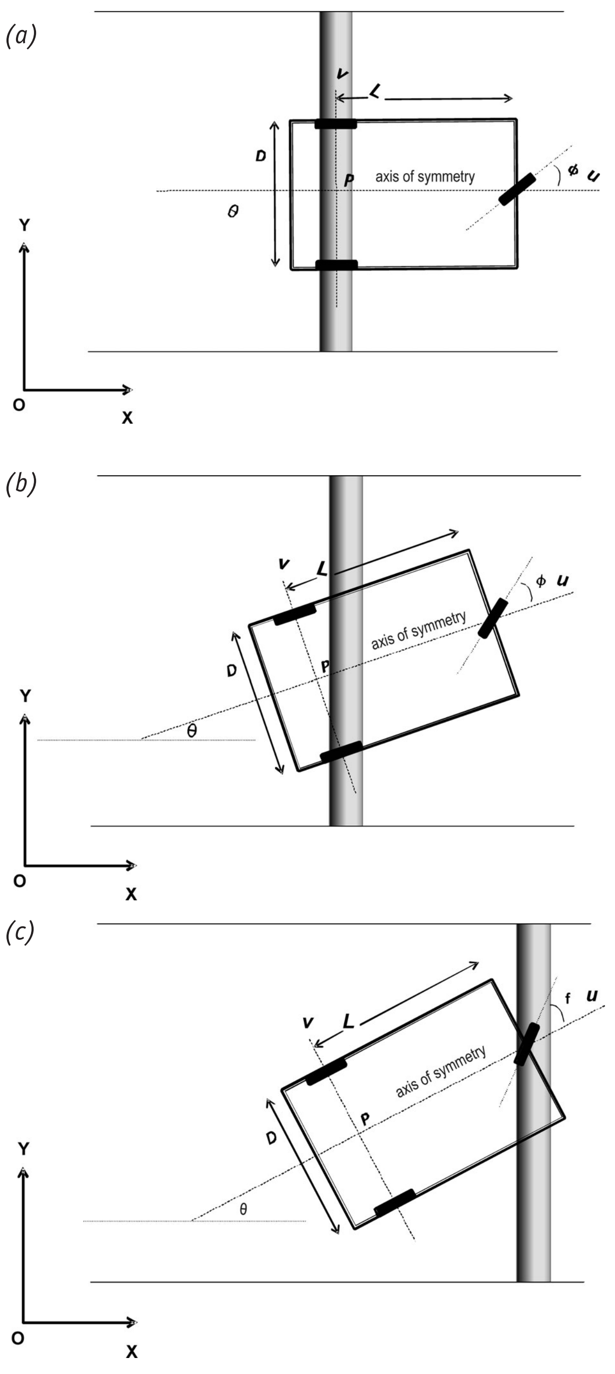

Fig. 4. Plan view of three typical postures of a vehicle over a hump shows an error condition of over counts when (a) both the rear wheels are over the hump, (b) when one of the rear wheels and (c) the front wheel is over the hump.

4. Implementation

The simplified functional block diagram of the error reduction system is shown in Fig. 5, which has two computation units and two switching units. The two computation units independently calculate the position and orientation incremental values by reading the corresponding encoder pulse counters from the encoder pulse processing modules. The switching units compare the incremental values and the least value will be selected for updating.

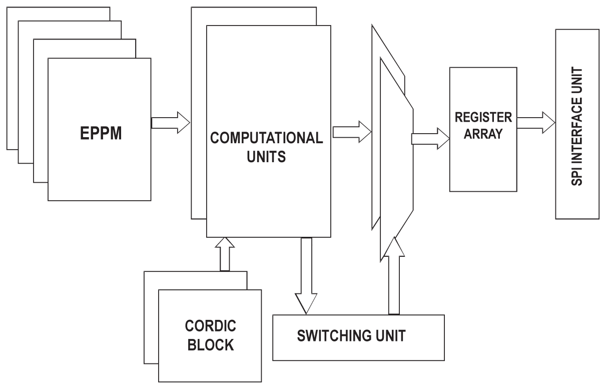

Fig. 5. The simplified functional diagram of the FPGA sub system, which computes two, sets of position increments and two attitude values. The switching, control and communication blocks are also shown.

The system implementation is achieved by using an Altera development board consisting of Cyclone-II EP2C70F672-C6 FPGA and associated components. The user interface termination provided on header connectors is used for realizing the system. The system is implemented with the help of the Quartus-II FPGA/CPLD design package and ModelSim simulation tools [27]. Fig. 5 shows the simplified functional block diagram of the system. Two-encoder pulse processing modules (EPPM) used to process the pulses from the rear wheel encoders place the velocity and position information in the appropriate registers. The other two encoders' pulses from the front driving steering wheel are not utilized for velocity calculations. The average velocity values measured from the rear wheels are available through the SPI interface of the system.

The computation unit solves the equations (1)-(6) for the calculation of position and attitude incremental update values. The sine and cosine calculation modules are necessary to speed up the process. So the outputs from the computation units are two sets of X axis and Y axis increments and two attitude values(θ). These values are stored in the corresponding registers and the switching module selects the least value set of position and attitude increments. All these values can be read through the SPI interface module.

4.1. The sine/cosine module

For the computation of Sine and cosine terms the famous CORDIC (COordinate Rotational DIgital Computer) algorithm is used. The CORDIC algorithm is an iterative technique based on the rotation of a vector, which allows many transcendental, and trigonometric functions to be computed. The highlight of this method is that it is achieved using only shifts, additions/subtractions and table look-ups, which map well with hardware and are ideal for FPGA implementation. The original work on CORDIC was carried out by Jack Volder [25] in 1959 for computing trigonometric functions as a part of developing a digital solution to real time navigation problems. Since then, much research has been carried out on this algorithm, with a thorough survey of this work with respect to FPGAs being published by Andraka [26].

All the trigonometric functions can be computed or derived from functions using vector rotations. The CORDIC algorithm is derived from the Givens rotation transforms:

$$x_{i+1} = x_i cos\theta(i) - y_i sin\theta(i)$$

$$y_{i+1} = y_i cos\theta(i) + x_i sin\theta(i)$$

which rotates a vector in Cartesian plane by an angle $\theta(i)$. These can be rearranged so that:

$$x_{i+1} = cos\theta(i)[x_i - y_i tan\theta(i)]$$

$$y_{i+1} = cos\theta(i)[y_i + x_i tan\theta(i)]$$

However the rotation angles $\theta(i)$ are restricted to $tan\theta(i) = \pm 2^{-i}$, then the multiplication by the tangent term is reduced to simple shift operation. Arbitrary angles of rotation are obtainable by performing a series of successive smaller elementary rotations. The iterative rotation can now be expressed as:

$$x_{i+1} = K_i[x_i - y_i \,.\, d_i \,.\, 2^{-i}]$$

$$y_{i+1} = K_i[y_i + x_i \,.\, d_i \,.\, 2^{-i}]$$

where:

$$K_i = cos(tan^{-1} 2^{-i}) = \frac{1}{\sqrt{1+2^{-2i}}}$$

$$d_i = \pm 1$$

Removing the scaling factor K_i from the iterative equations yields a shift-add algorithm for vector rotation. The product of K_i's can be applied elsewhere in the system. The product approaches a constant value as the number of iterations goes to infinity. The exact gain of the system depends on the number of iterations and is:

$$A_n = \prod_n \sqrt{1+2^{-2i}}$$

The set of all possible decision vectors in an angular measurement system is based on binary arctangents. Conversions between this angular system and the other can be accomplished with the help of a look-up table, which can be easily implemented in an FPGA. The angle accumulator adds a third difference equation to the CORDIC algorithm:

$$z_{i+1} = z_i - d_i \,.\, tan^{-1}(2^{-i})$$

The CORDIC rotator is normally operated in one of the two modes. The first, called *rotation mode*, rotates the input vector by a specified angle and the second mode called *vectoring mode* rotates the input vector to the X axis while recording the angle required to make that rotation.

In this system the rotation mode is used to compute the sine and cosine values to solve the position and attitude update equations. In this mode the angle accumulator is initialised with the desired rotation angle. The rotation decision at each iteration is made to diminish the magnitude of the residual angle in the angle accumulator. The decision at each iteration is therefore based on the sign of the residual angle after each step. For rotation mode the CORDIC equations are:

$$x_{i+1} = x_i - y_i \,.\, d_i \,.\, 2^{-i}$$

$$y_{i+1} = y_i + x_i \,.\, d_i \,.\, 2^{-i}$$

$$z_{i+1} = z_i - d_i \,.tan^{-1}(2^{-i})$$

where
$d_i = -1$ if $z_i < 0$, $+1$ otherwise

Which provides the following results:

$$x_n = A_n[x_0 \cos z_0 - y_0 \sin z_0]$$

$$y_n = A_n[y_0 \cos z_0 + x_0 \sin z_0]$$

$$z_n = 0$$

$$A_n = \prod_n \sqrt{1+2^{-2i}}$$

The rotation mode CORDIC operation can simultaneously compute the sine and cosine of the input angle. By setting $x_0 = 1\,; y_0 = 0\,; z_0 =$ *the input angle*; and multiplying the result with $(1/A_n)$ the sine and cosine values can be computed. Fig. 6 shows the angle error plot against the variations in residual angle with respect to the number of iterations. The value of $1/A_n$ for 16 iterations is around 0.60725293510314. More than ten iterations give a satisfactory resolution and $1/A_n$ is nearly a constant.

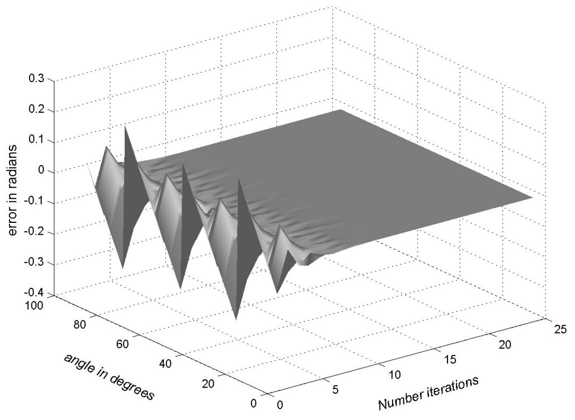

Fig. 6. The 3D plot showing the variations in residual angle, which is a measure of computational error against input angle and, the number of iterations.

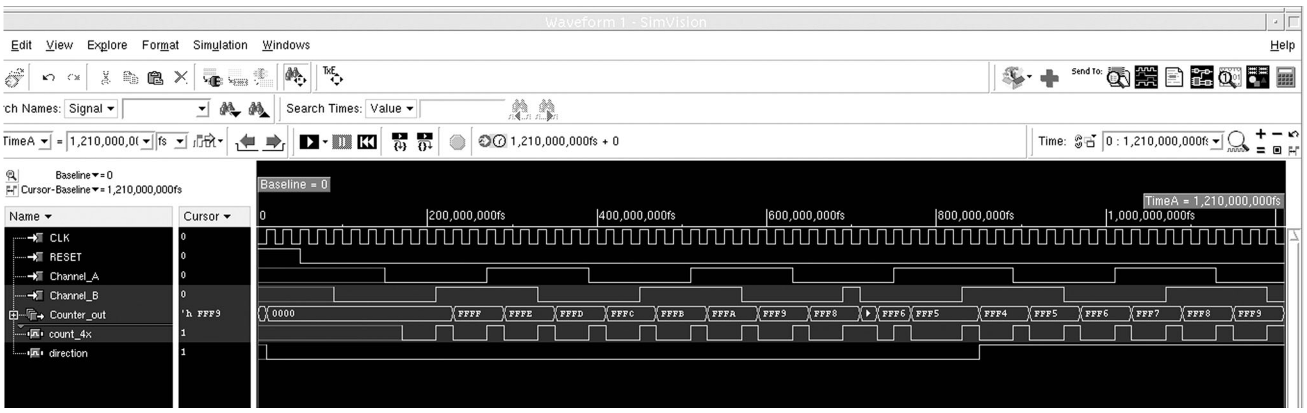

Fig. 7. Screen shot showing the various waveforms with system clock of 50MHz of encoder pulse processing module.

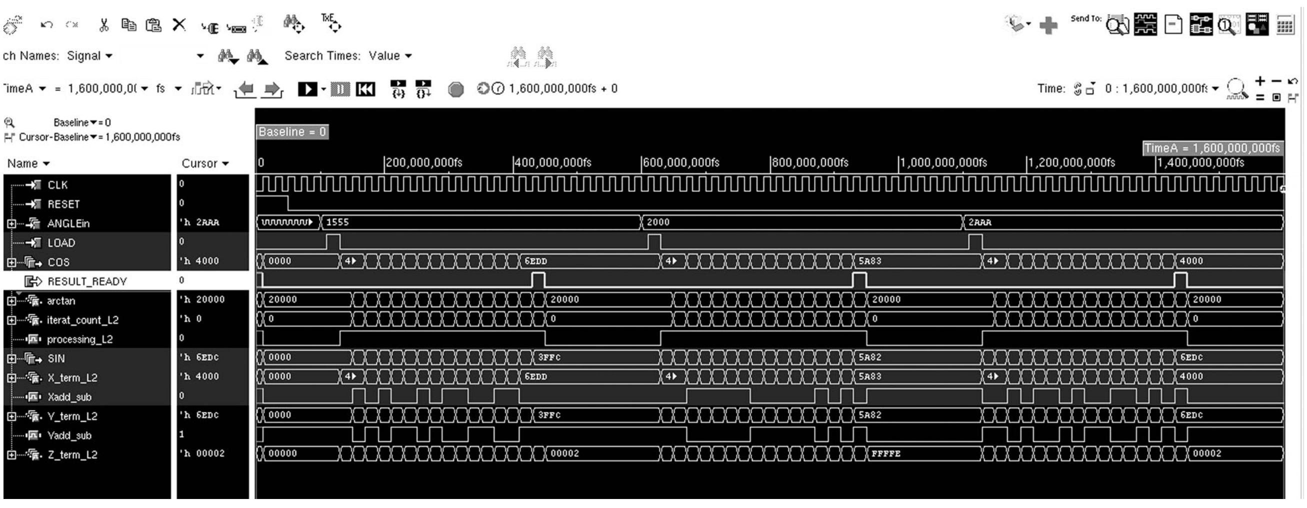

Fig. 8. Screen shot showing the various results of computation along with the control and status signal associated with the sequential CORDIC module.

4.2. The Realization Details and Results

The entire sub system has been designed and implemented in Altera Cyclone-II EP2C70F672-C6 FPGA. The main task is the implementation of the computational unit consisting of the CORDIC processors for the sine and cosine calculation. The design has been divided into various blocks like the Encoder Pulse Processing Module (EPPM), computation unit for solving the update equations incorporating the CORDIC processor, switching module for selecting least values and SPI communication interface for establishing the link with the control processor. There are various read/write registers involved in the design for scaling, initialisation and data storing.

The entire design entry process is carried out with the help of VHDL and the design is synthesized with the above mentioned target device in Quartus II software. The CORDIC algorithm is implemented in sequential manner and the number of iterations is 16. The sequential CORDIC design performs one iteration per clock cycle. With a system clock frequency of 50 MHz, the total time taken by this module is less than 400 ns, which is a very small value as most of the vehicle control system require only one sample per 100 µs or less. So the results are ready with the sub system after initialisation of the system registers and the role of the controller is to read the update values and velocity information through the SPI. Fig. 7 shows the screen shot during simulation of the Encoder pulse-processing module.

Fig. 8 shows the various waveforms during calculation of sine and cosine values of 30 degrees, 45 degrees and 60 degrees. The inputs to the CORDIC module ANGLEin and LOAD signals are pulsed and RESULT_READY indicates the availability of the results in the 16- bit registers after sixteen clock pulses. The photograph of the sub system is shown in Fig. 9.

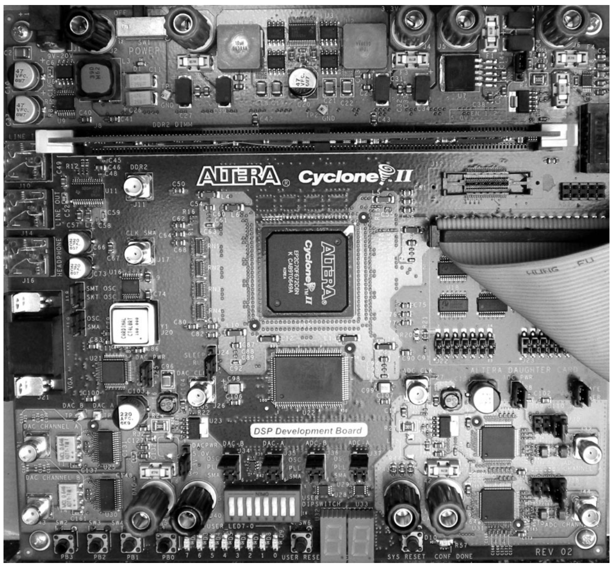

Fig. 9. Photograph of the sub system implemented in Altera Cyclone-II EP2C70F672-C6 - FPGA development board.

5. Conclusions

An efficient and novel technique to reduce the odometric error and the implementation details of the same in an FPGA for reducing the odometric localization errors caused by over count readings of an optical encoder based odometric system in a mobile robot due to wheel-slippage and terrain irregularities has been presented in this paper. The standard quadrature technique is used to obtain four counts in each encoder period. By using this system one can reduce the odometric error so as to increase the travel distance between absolute position updates thereby lowering the installation and operating costs for the system. For the velocity measurement unit, a change in the direction bit inside a time window may cause error. This should be eliminated. With the use of this sub system, most of the odometric computational burden can be released from the control processor associated with the mobile robot vehicle. The uneven loading on rubber wheels which, most of the mobile robot vehicles use, may cause erroneous readings in count values due to change in wheel diameter. Even a tilt of the vehicle may cause uneven loading. This approach may not be recommended for mobile robot vehicles having chances of frequent skidding during path execution. Under most situations this approach is quite satisfactory, simple and efficient.

AUTHORS
James Kurian*, P.R. Saseendran Pillai - Department of Electronics, Cochin University of Science and Technology, Cochin-682022, Kerala, INDIA.
E-mail: james@cusat.ac.in.
* Corresponding author

References

[1] Siegel M., Gunatilake P., "Remote Inspection Technologies for Aircraft Skin Inspection". In: *IEEE Workshop on Emergent Technologies and Virtual Systems for Instrumentation and Measurement, Niagara Falls,* Ontario, Canada, 15th-17th May 1997, pp. 69-78.

[2] Prassler E., Ritter A., Schaeffer C., Fiorini P., "A Short History of Cleaning Robots", *Autonomous Robots, Special Issue on Cleaning and Housekeeping Robots,* vol. 9, issue 3, December 2000.

[3] Iborra A., Pastor J., Alvarez B., Fernandez C., Merono J., "Robots in Radioactive Environments", *IEEE Robotics and Automation Magazine,* vol. 10, no.4, December 2003, pp. 12-22.

[4] Anthony Stentz, John Bares, Sanjiv Singh and Patrick Rowe, "A Robotic Excavator for Autonomous Truck Loading", *Autonomous Robots,* vol. 7, 1999, pp. 175-186.

[5] Rossetti M. D., Kumar A. and Felder R., "Mobile Robot Simulation of Clinical Laboratory Deliveries". In: *Proceedings of the 30th Conference on Winter Simulation,* Washington, D.C., United States, 1998, pp. 1415-1422.

[6] Montemerlo M., Pineau J., Roy N., Thrun S., Verma V., "Experiences with a Mobile Robotic Guide for the Elderly". In: *Proceedings of the 18th AAAI National Conference on Artificial Intelligence,* Edmonton, Canada, 2002, pp. 587-592.

[7] Bahl R., "Object Classification using Compact Sector Scanning Sonars in Turbid Waters". In: *Proc. 2nd IARP Mobile Robots for Subsea Environments,* Monterey, CA USA, vol. 23, 1990, pp. 303-327.

[8] Healey A.J., "Application of Formation Control for

Multi-Vehicle Robotic Minesweeping". In: *Proceedings of the 40th IEEE Conference on Decision and Control,* vol.2, 2001, pp. 1497-1502.

[9] Everett H.R., Gage D.W., "From Laboratory to Warehouse: Security Robots Meet the Real World", *International Journal of Robotics Research, Special Issue on Field and Service Robotics,* vol. 18, no. 7, July 1999, pp. 760-768.

[10] Pastore T.H., Everett H.R., and Bonner K., "Mobile Robots for Outdoor Security Applications", *American Nuclear Society 8th International Topical Meeting on Robotics and Remote Systems (ANS'99),* Pittsburgh, PA, USA April 1999. Available at: http://handle.dtic.mil/100.2/ADA422047

[11] Borenstein J., Everett H., Feng L., Wehe D., "Mobile Robot Positioning Sensors and Techniques", *Journal of Robotic Systems, Special Issue on Mobile Robots,* vol. 14, no. 4, 1997, pp. 231-249.

[12] Kleeman L., "Optimal Estimation of position and Heading for Mobile Robots Using Ultrasonic Beacons and Dead-reckoning". In: *Proceedings of the IEEE International Conference on Robotics and Automation*, Nice, France, 1992, pp. 2582-2587.

[13] Leonard J.J., Durrant-Whyte H.F., "Mobile Robot Localization by Tracking Geometric Beacons", *IEEE Transactions on Robotics and Automation,* vol. 7, no. 3, 1991, pp. 376-382.

[14] Wijk O., Christensen H.I., "Localization and navigation of a mobile robot using natural point landmarks extracted from sonar data", *Robotics and Autonomous Systems,* vol. 31, 2000, pp. 31-42.

[15] L´Opez-s´ Anchez M., Esteva F., L´Opez De M`Antars R., Silerra C., "Map Generation by Cooperative Low-Cost Robots in Structured Unknown Environments", *Autonomous Robots,* vol. 5, 1998, pp. 53-61.

[16] Sukkarieh S., Nebot E.M., Durrant-Whyte H.F., "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle Applications", *IEEE Transactions on Robotics and Automation,* vol. 15, no. 3, 1999, pp. 572-578.

[17] Borenstein J., Feng L., "Measurement and Correction of Systematic Odometry Errors in Mobile Robots", *IEEE Transactions on Robotics and Automation*, vol.12, no. 6, 1996, pp. 869-880.

[18] Barshan B., Durrant-Whyte H.F., "Inertial Navigation Systems for Mobile Robots", *IEEE Transactions on Robotics and Automation*, vol.11, no. 3, 1995, pp. 328-342.

[19] Lee S., Song J.-B., "Robust Mobile Robot Localization using Optical Flow Sensors and Encoders". In *Proc. of the IEEE Int. Conf. on Robotics & Automation*, April 2004, pp. 1039-1044.

[20] De Cecco M., "Sensor fusion of inertial-odometric navigation as a function of the actual manoeuvres of autonomous guided vehicles", *IOP Meas. Sci. Technol.,* vol.14, 2003, pp. 643-653.

[21] Faccio M., *et al.*, "An embedded system for position and speed measurement adopting incremental encoders". In: *Proc. of the IEEE Ind. Appl. Conf.,* vol. 2, 2004, pp. 1192-1199.

[22] Ekekwea N., Etienne-Cummingsa R., and Peter Kazanzidesb, "A wide speed range and high precision position and velocity measurements chip with serial peripheral interface", *ELS. INTEGRATION, the VLSI journal,* vol. 41, 2008, pp. 297-305.

[23] Hebert B., Michel Brule M., Dessaint L.-A., "A High Efficiency Interface for a Biphase Incremental Encoder with Error Detection", *IEEE Transactions on Industrial Electronics*, vol. 40, no. 1, 1993, pp. 155-156.

[24] Ojeda L., Borenstein J., "Reduction of Odometry Errors in Over-constrained Mobile Robots". In: *Proceedings of the UGV Technology Conference at the 2003 SPIE Aero-Sense Symposium,* Orlando, FL, USA, 21st-25th April 2003, vol. 5083, pp. 431-439.

[25] Volder J., "The CORDIC Trigonometric Computing Technique", *IRE Trans Electronic Computing,* vol. EC-8, September 1959, pp. 330-334.

[26] Andraka R., "A survey of CORDIC algorithms for FPGA based computers". In: *Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field Programmable Gate Arrays,* 22nd-24th February, 1998, pp. 191-200.

[27] Altera Corporation, "Cyclone II DSP Development Board Reference Manual", *101 Innovation Drive,* San Jose, CA 95134, USA. http://www.altera.com