Slope-Based Point Pursuing Manoeuvers of Nonholonomic Robots using FPGA

Ying-Hao Yu, S. Kodagoda, and Q.P. Ha

***Abstract*— Steering maneuver is essential in robotic motion planning. Despite a lot of steering mechanisms successfully developed in past years, for miniature robots, real-time computation is still a limitation for robot path tracking. The design issues in cooperative control of battery-powered nonholonomic robots rest with the complicacy of the control strategies, the low power consumption and real-time processing capability. Conventionally, the improvement of computing speed mostly relies on the increment of the system clock and often results in some transient loss. Thus, an elaborate control algorithm developed for PC might not work on an embedded system. This paper presents a comprehensive steering algorithm which, via issuing suitable predicaments for computation, will dramatically reduce the resource usage in hardware circuit design. The proposed algorithm is implemented on an embedded system using the field programmable gate array (FPGA) technology.**

I. INTRODUCTION

THE most popular robotic steering systems of wheeled robots are the differential-drive and Ackermann steering. The motion of the differential-drive robots are controlled by individually controlling the motors driving each wheel. The same speed of motors produces straight line motion whereas different speeds cause the robot to turn. On the other hand, the Ackermann steering vehicles use two separate actuators for driving and turning [1].

Steering control of non-holonomic robots in indoor environments can be modelled by using the space coordinates such as the turning angle, angular turning speed, and orientation of robots due to their relatively slow speed of motion [2]. The lateral sliding influence is not as significant as a high speed car [3]. During an autonomous driving scenario, two essential parameters can be considered for steering strategies. The first parameter is called the look-ahead distance, i.e. the specified virtual distance in front of the robot. In real-world applications, the specified look-ahead distance helps a vehicle to decide the deviation from the central line of road [4], or it can also be extended to calculate accordingly the speed with respect to ground [5]. The second critical parameter is the pursued point, which is the destination of a look-ahead distance. Consequently, a long and sweeping route in the cruising space can be composed of sequential pursuing points, and the expected trajectory of robot is achieved by different angles of turning. If we assume a wheeled indoor robot is tracking on planned points, the shortest path tracking will depend on the least swing of every look-ahead path.

There have been many techniques successfully developed in past years to obtain optimal paths for differential-drive and car-like vehicle steering [6,7]. Minimizing errors in vehicle path tracking can be achieved via the use of rigorous control techniques [4,5]. However, autonomous steering requires the availability of measurable parameters from the robots and cruising space, resulting often in much computing effort for a higher accuracy. In an indoor environment, real-time computation remains a limitation for robot path tracking with miniature robots. The complicacy of the control strategies, the low power consumption and real-time processing capability are design issues for cooperative control of battery-powered nonholonomic robots. When implementing complicated strategies for vehicle control on an embedded system, the overall motion may also inherit the problem of tracking stability. This problem has been discussed in [8], revealing the instability of steering control with short horizons, and is generally ascribed to the requirement of a drastic reaction to a sharp incident angle for error correction to maintain the desired path. Reactive tracking for a group of robots is proposed in [9] using the variable structure methodology, but also facing implementation difficulties. The tracking stability problem could be alleviated by employing a larger look-ahead distance. Unfortunately, such long look-ahead schemes may not be realistic in an indoor environment where the driving space is limited.

In contrast to control theoretic solutions, the pure-pursuit task executing the shortest path between two points can be realized only with a single turning [10]. Although the pure-pursuit algorithm is an efficient mechanism to reach the expected destination, it does not guarantee the orientation of the robot with that of the path at the destination [11]. The problem is originated from the curve steering nature which requires to always keep an incident angle to path. Towards a valid solution, the use of a behavior-based model for robot steering seems to be feasible with small values of speed and acceleration of a robot

This work was supported by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

The authors are with the ARC Centre of Excellence for Autonomous Systems (CAS), Faculty of Engineering and Information Technology, University of Technology, Sydney, PO Box 123 Broadway NSW 2007 Australia. (E-mail: {YingHao.Yu|sakoda|quangha}@eng.uts.edu.au).

navigating in an indoor environment. The steering behavior can be represented by geometric representations which have been used to coin the shortest path planning in mobile robotics. Dubins first estimated the shortest path between two points in an obstacle free space by combining clockwise turning (R), anticlockwise turning (L), and a straight line driving (S), e.g. RSL or LSR maneuvers [12]. Although it proposed a good paradigm in behavior-based steering, finding the start and end point of a tangent between two turning arcs is not an easy task for embedded systems [13].

From [7], one can see that the shortest path for nonholonomic robots in an obstacle free environment comprises in general a trajectory combining three labels. Motivated by this framework and Dubins' turning mechanisms, we introduce in this paper a slope-based are-line-arc (SBALA) algorithm, aiming at implementation of robot steering on a programmable chip for the shortest path. Notably, this algorithm involves reduced computing effort, and hence allowing the steering maneuvers to be realized directly on an embedded system with hardware circuit design.

The remainder of this paper is organized as following: Section II describes the SBALA geometric algorithm. Section III provides the development of the proposed steering mechanism on an FPGA kit for a differential-drive platform, the Eyebots. Experimental results and discussion are included in Section IV. Finally, Section V concludes the paper.

II. Slope Based Arc-Line-Arc Algorithm

A great deal of research has been devoted to the problem of planning collision-free optimal trajectories for nonholonomic mobile robots that move forward only [12] or move forward and backward [14], whereby the shortest paths comprise straight line segments and arcs subject to bounded turning radius [6].

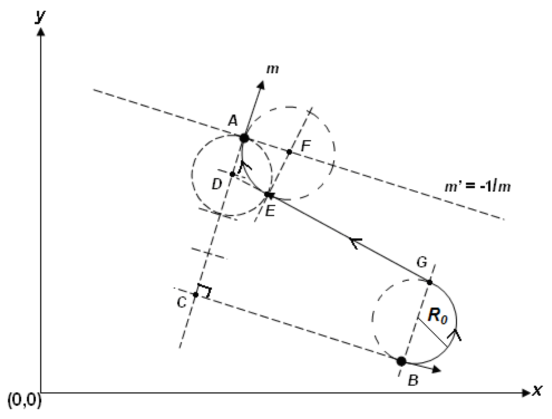


Fig. 1. A representation of SBALA on 2D plane.

A. SBALA Algorithm

By considering the least computing effort and resource usage on embedded systems, particularly for hardware circuit design, the angular and trigonometric representations are firstly replaced by the instant tangent slope of the robot path on 2D plane. Figure 1 shows an example of the slope based arc-line-arc algorithm (SBALA) in 2D representation.

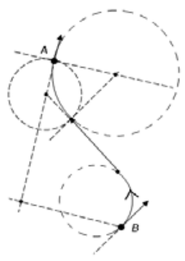
The SBALA is mainly composed of seven critical points. Point *A* is the expected destination to pursue, and *B* is the central point of the robot at its initial location. Slope *m* represents the trajectory tangent at point *A* in (*x*,*y*)-coordinates, and another slope perpendicular to the trajectory at *A* is denoted as *m'*. The look-ahead distance in Fig. 1 is the straight distance between point *A* and *B*. Point C is the perpendicular intersection point from *B* and the desired orientation at destination *A*. To provide the turning reference of the robot, $\overline{AC}$ is separated into four equal sections, and R_0 is the turning radius $1/4\,\overline{AC}$ at point *B*. More division numbers may be used for differential-drive robots while a section length will be limited by the maximum turning curvature of a vehicle-like robot. The orientation of the robot at *B* in Fig. 1 is conveniently set aligned to $\overline{BC}$.

Next the robot starts to pursue the point *A* with an initial anticlockwise turning (L). Line $\overline{DG}$ is the tangent of the robot's first turning arc. After the robot has reached point *G*, it starts a straight line driving (S) toward point *D* which is located at $1/4\,\overline{AC}$. Point *E* is located at the ring which circles point *D* with radius $1/4\ \overline{AC}$. Once the robot reaches point *E*, the second arc can be derived with radius $\overline{AF} = \overline{EF}$, completing the path *B* to *A* with an aligned orientation.

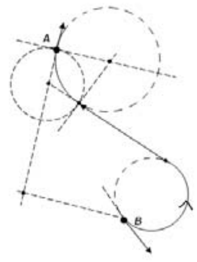
B. General Cases of SBALA Algorithm

Figure 2 shows four possible SBALA maneuvers using R, S and L combinations. The LSR maneuvers with different trajectories of the robot for different initial orientations at point B are shown from Fig. 2(a) to (c), and a RSL maneuvers is shown in Fig. 2(d). Those diagrams show realistic paths to the pursuing point. On the straight path (c), the curvature of the second arc will be approximated to zero, so the robot nearly keeps a straight line for driving on the look-ahead path.

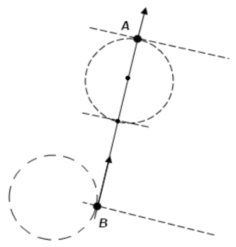
From Fig. 2, a question may be asked as to whether the trajectory using SBALA algorithm is the shortest turning path. Since the maneuvers of the straight line and second arc driving are dominated by the first arc, this is the problem of finding the shortest path for the first arc toward point *D*, then determining the turning by LSR or RSL maneuver, see Fig. 3. The shortest turning rules for arbitrary initial orientations of the robot are summarized in Table I by comparing the slopes of the straight line $\overline{BD}$ and at the starting point *B*.



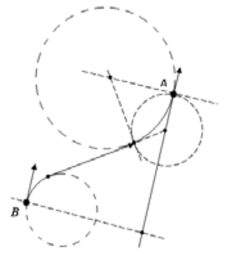
(a) LSR, small pursuing angle.



(b) LSR, large pursuing angle.



(c) LSR on straight path driving.



(d) RSL on left side of $\overline{AC}$.

Fig. 2. Possible cases of SBALA algorithm.

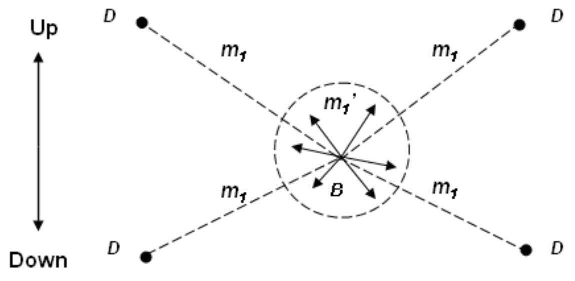


Fig. 3. Point pursuing for different locations of point D.

Table I
The shortest turning rules for the first arc toward to point *D*.

m_1	m_1'	$m_1 - m_1'$	$D_{(y)} \geq B_{(y)}$		$D_{(y)} < B_{(y)}$	
			Up	Down	Up	Down
+	+	≥ 0	L	R	R	L
		< 0	R	L	L	R
-	-	≥ 0	L	R	R	L
		< 0	R	L	L	R
-	+	< 0	L	R	R	L
+	-	> 0	R	L	L	R

In Table I, m_1 is the slope of $\overline{BD}$, and m_1' is the slope of at point *B* for an arbitrary initial orientation of the robot. Notation "Up" is for the robot is moving forward on 2D plane while "Down" is for the reverse direction. $D_{(y)}$ and $B_{(y)}$ represent the coordinates in *y* at *D* and *B*. The turning rules on Table I will inverse with different longitudinal driving directions or when point D crosses the horizontal axis corresponding polar angle 0 or π.

III. SBALA Implementation on FPGA

The FPGA platform used in our study is the Altera DE2-70 equipped with Cyclone II which is shown in Fig. 4(a). The camera utilized is a 5-Mega pixel digital camera module from Terasic shown in Fig. 4(b) with shutter speed of 34fps and 1024x1280 pixel resolution. In the test scenario, the monocular digital camera is used as the environment mounted global camera to track and control two differential-drive robots of Eyebot type shown in Fig. 4(c).

For FPGA implementation of the proposed SBALA algorithm, we have utilized our previous designs for machine vision. The first design is the color discrimination function [15,16]. Instead of using computationally demanding pattern recognition algorithms, the FPGA detects the dual color of bull eye labels on the top of Eyebots. The outer green ring of the bull eye is used for interference reduction, and the inner blue area is the interested label. The FPGA tracks Eyebots (labels) which start off from a docking area on the bottom of the monitor screen via specified threshold of blue pixel numbers. The second design is the relative distance estimation algorithm based on Perspective Projection Image Ratio (PPIR) [17]. Instead of performing relative distance measurement using the sensors installed on robots, we use a monocular global camera to estimate the relative distance between labels on the robots. Further, it provides the ratio of the real distance and the diameter of labels (circles), i.e. distance in ratio equal real distance over circle diameter.

Once the color discrimination and PPIR relative distance estimation functions have been implemented on the FPGA platform, the coordinates of the leader robot, follower robot, and pursed point can be determined. Fig. 5 depicts the test scenario requiring the robot located at bottom of the figure (*B*) to pursue and align with the virtual leader robot on the top of the figure (*A*). Firstly, we deliberately define the central point of a leader's label at $L(x_0,y_0) = L(40,40)$ in the global coordinates for PPIR distance ratio. This value is decided by the ratio of maximum view range of the camera over the diameter of circular label. The instant trajectory tangent (slope) *m* is measured by integrated movement of 1/5 label length. Once the pursuance has started, the Eyebots move forward shortly then stop, and the slopes of trajectories on 2D plane are accordingly recorded by FPGA.

(a) FPGA platform with Cyclone II FPGA.

(b) digital camera.

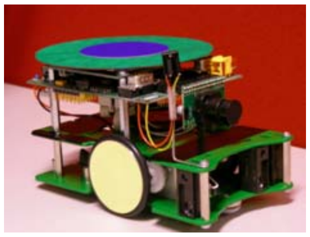

(c) Eyebot with bull eye label.

Fig. 4. Devices for SBALA implementation.

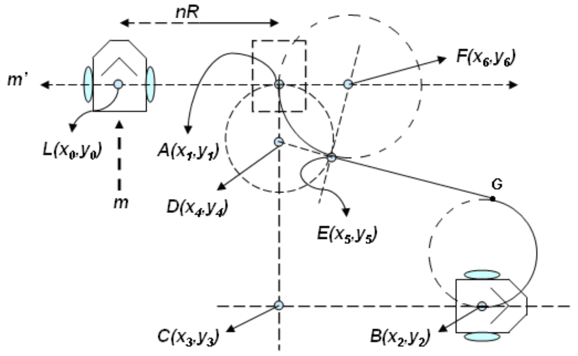


Fig. 5. Test scenario for SBALA algorithm

In the next step, the follower robot starts to pursue the expected point with a constant speed and driving distance (curve length) in every remote instruction received, then FPGA continuously sends driving angles in radians to Eyebot for turning. The pursued point $A(x_1,y_1)$ can be derived by FPGA as:

$$m' = \frac{-1}{m} = \frac{(y_1 - y_0)}{(x_1 - x_0)}, \tag{1}$$

where its coordinates are obtained from

$$(x_1 - x_0)^2 + (y_1 - y_0)^2 = (nR)^2 \ , \tag{2}$$

as

$$x_1 = x_0 \pm \frac{nR}{\sqrt{1+(m')^2}} = x_0 \pm X,$$
$$y_1 = y_0 \pm \frac{nR}{\sqrt{1+\left(\frac{1}{m'}\right)^2}} = y_0 \pm Y, \tag{3}$$

in which the sign of X and Y are respectively decided by the location of the pursued point and slope m, R is the circular label diameter, n is the multiple of label's diameter, and m' is the perpendicular slope to the leader's orientation. The location of follower robot $B(x_2, y_2)$ is decided by the PPIR relative distance to the global point L. Similarly, point $C(x_3,y_3)$ in Fig. 5 is derived from slopes:

$$\frac{(y_1 - y_3)}{(x_1 - x_3)} = m, \tag{4}$$

and

$$\frac{(y_2 - y_3)}{(x_2 - x_3)} = m' \tag{5}$$

as

$$x_3 = \frac{y_2 - y_1 - m'x_2 + mx_1}{m - m'}$$
$$y_3 = y_1 - mx_1 + mx_3. \tag{6}$$

Point $D(x_4,y_4)$ is determined according to the SLABA algorithm as:

$$x_4 = \frac{3}{4}(x_1 - x_3) + x_3$$
$$y_4 = \frac{3}{4}(y_1 - y_3) + y_3. \tag{7}$$

The radius of first arc (assumed left turn, L) is set at $R_0 = 1/4\ \overline{AC}$ in the proposed algorithm. For every turning instant, if following robot's trajectory tangent is matched within the tolerance of the $\overline{BD}$ slope, point G will be recorded, then the follower robot starts the segment of straight driving (S). As soon as the mobile robot reaches the circle around point D at distance of about $1/4\ \overline{AC}$ ($\overline{BD} \approx 1/4\ \overline{AC}$), point E is then determined. The center, point $F(x_6,y_6)$, can be derived via coordinates of A, D, and E and slopes

$$\frac{(y_6 - y_1)}{(x_6 - x_1)} = m', \tag{8}$$

$$m_2 = \frac{(y_5 - y_4)}{(x_5 - x_4)} = \frac{(x_6 - x_5)}{(y_5 - y_6)}, \tag{9}$$

as

$$x_6 = x_5 + m_2 y_5 - m_2 y_6,$$
$$y_6 = \frac{y_1 - m'x_1 + m'x_5 + m'm_2 y_5}{1 + m'm_2}. \tag{10}$$

The robot is then driving along the second arc (R) with radius $AF = EF$.

By taking into account possible motion sliding factors, the real driving SBALA strategy is implemented on the FPGA platform with the following pseudo codes:

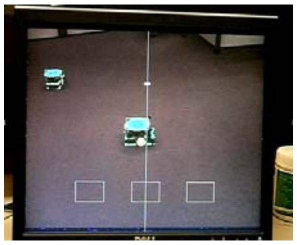
```
If slope($B$) ≠ slope($\overline{BD}$) then
  reload arc_1

Else
  record point $G$
  load straight line drive "S"

If straight line drive
   If |slope($B$) - slope($\overline{DG}$)| > tolerance then
        reload  arc_1
        clear point $G$

   If point $E$ then
      load arc_2

If destination $A$ then
  stop or reload arc_1
```

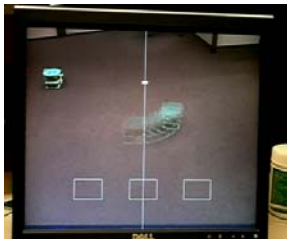
All the other functions including raw image demosaicing, color discrimination, relative distance estimation, target tracking, and cruising control with infrared signal are also implemented on the FPGA chip using pure hardware circuit designs.

IV. Experimental Results and Discussion

The real-time tracking image from the digital camera is shown on a computer monitor through the VGA interface with the FPGA platform. Figure 6(a) to (f) demonstrates the performance of the follower robot in achieving its final destination. The results show satisfactory establishment of into a row formation.

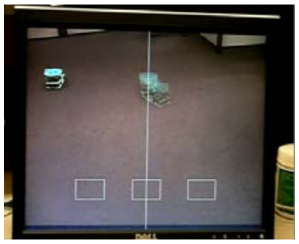
Notably, since the proposed SBALA algorithm is developed based on Dubins' turning rules with LSR and RSL maneuvers to suite the requirement of FPGA implementation, the start and terminal points of the tangent between arcs are determined by using just basic arithmetic operations. Another interesting feature is that SBALA compromises the absolutely shortest tracking path of the second arc in calculating with slopes. Also, it makes the final turning less abrupt than by using Dubins' rules. Such design has the advantage for easing off the error from motion sliding during the final turning process. In our experiments, the SBALA operations can cope with large incident angles. The driving path is scheduled with seven points leading to smooth tracking, so the steering error is naturally converged. Even driving on a straight path, the proposed algorithm is still satisfied with zero or minute curvature of first and second arcs. Contrarily, the stability conditions in control techniques may require a small incident angle to avoid over steering control from drastic error correction.


(a) Initial deployment


(b) Follower robot starts first arc turning.

(c) Continuing first arc turning.

(d) Running on straight driving.


(e) Running on second arc.

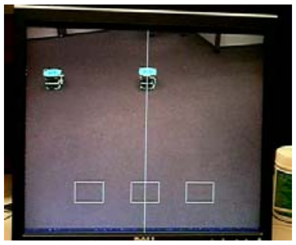

(f) Reaching destination.

Fig. 6. Point pursuing for row formation with two Eyebots.

Finally, the requirements of real-time and low power consumption on embedded systems are also considered in the proposed SBALA algorithm. Although both issues have been overcome via hardware circuit designs, unfortunately, the floating point operations with trigonometric functions consume too many logic gates, so it is impossible to implement directly in hardware circuitry with a programmable embedded system [18]. Here, the SBALA is designed suitably to interpret robots' trajectory tangent in 2D slope with the purpose to reveal the possibility of realizing the steering control via basic binary operations. For example, by incorporating the PPIR algorithm [17], a smaller space unit can be represented in ratios with respect to the label diameter. With a reasonable error for several percents, it accepts numbers which are simply re-scaled by multiplying for 100, and the square rooting problem can be also easily approximated by using the following algorithm [19]:

$$\sqrt{x^2+y^2} \cong \max(((a-0.125a)+0.5b),a), \tag{11}$$

where $a = \max(|x|, |y|)$, and $b = \min(|x|, |y|)$, with values 0.5 and 0.125 being expressed by right shift operations for 1 and 3 bits. All of these approximations lead to the successful implementation for robotic steering control via the register transfer level (RTL) circuits on a programmable embedded system.

V. Conclusion

We have presented an effective for robot steering with its implementation on a programmable chip. By improving Dubins' turning maneuvers of, we have solved the steering problem for robotic path tracking with minimum design effort. This innovative feature can contribute to the required real-time implementation with low power computation in embedded systems. According to the experimental results, the proposed slope-based arc-line-arc algorithm demonstrates its comparable abilities in stable steering and path tracking. By breaking away from the concept of trigonometric modeling, the algorithm can model the steering maneuvers by trajectory tangents in 2D. Consequently, the operations of floating point become an option in such algorithm. These advantages make the proposed SBALA algorithm feasible and efficient for implementation on embedded systems, particularly for the application of the FPGA technology in ubiquitous robotics.

References

[1] T. Bräunl. *Embedded Robotics*. Springer, Australia, 2006, pp. 97-121.

[2] B. Li and C. Zhang," Adaptive fuzzy control for mobile robot obstacle avoidance based on virtual line path tracking," *Proc. IEEE Intl. Conf. Robotics and Biomimetics*, Kunming, China, 2006, pp.1454-1458.

[3] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. USA: SAE, 1992, pp. 196–235.

[4] R. Marion, S. Scalzi, G. Orlando, and M. Netto, "A Nested PID Steering Control for Lane Keeping in Vision Based Autonomous Vehicles," *Proc. American Control Conf.*, Missouri, USA, 2009, PP. 2885-2890.

[5] Y. Hayakawa, R. White, T. Kimura, and G. Naitou," Driver Oriented Path Following in ITS," *Proc. IEEE Intl. Conf. Advanced Intelligent Mechatronics*, Illinois, USA, 2003, vol. 1, pp.558-563.

[6] S. Bhattacharya, R. Murrieta-Cid, and S. Hutchinson, "Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints," *IEEE Trans. on Robotics*, Vol. 23, pp. 47–59, 2007.

[7] F. Lamiraux and J.-P. Laumond, "Smooth Motion Planning for Car-Like Vehicles," *IEEE Trans. on Robotics*, vol. 17, pp. 498-502, 2001.

[8] S.-B. Wu, H-Y. Chen, and M-Q. Zheng, "Study on Stability for Steering Closed-Loop System of Remote-Operated Tracked Vehicle," *Proc. IEEE Intl. Conf. on Machine Learning and Cybernetics*, Montreal, Canada, vol.6, 2009, pp.3145-3149.

[9] Q.P. Ha and G. Dissanayake, "Robust Formation of Multiple Robots using Reactive Variable Structure Systems," *Int. Trans. Systems Science and Applications*, Vol. 1, No. 2, pp. 183-192, 2006.

[10] J. Morales, J-L. Martínez, A. Martínez, and A. Mandow," Pure-Pursuit Reactive Path Tracking for Nonholonomic Mobile," *EURASIP Journal on Advances in Signal Processing*, Article ID 935237, 10 pages, vol. 2009.

[11] Y. Suzuki, S. Kagami, and J. J. Kuffner, "Path Planning with Steering Sets for Car-Like Robots and Finding an Effective Set," *IEEE Intl. Conf. on Robotic and Biomimetics*, Kunming, China, 2006, pp.1221-1226.

[12] L.E. Dubins, "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, Vol. 79, pp. 497-516, 1957.

[13] A. Balluchi, A. Bicchi, A. Balestrino, and G. Casalino, "Path Tracking Control for Dubin's Cars," *Proc. IEEE Intl. Conf. on Robotic and Automation*, Minnesota, USA, 1996, Vol. 4, pp. 3123-3128.

[14] J.A. Reeds and L.A. Shepp, "Optimal paths for car that go both forwards and backwards," *Pacific Journal of Mathematics*, Vol. 145, pp. 367-393, 1990.

[15] Y.-H. Yu, N.M. Kwok, and Q.P. Ha, "FPGA-based Real-time Color Discrimination Design for Ubiquitous Robots" *Proc. Australasian Conference on Robotics and Automation*, Sydney, Australia, 2009.

[16] Y.-H. Yu, N. M. Kwok, and Q. P. Ha, "Chip-based Design for Real-time Moving Object Detection using Digital Camera Module," *Proc. of the 2nd International Congress on Image and Signal Processing* (CISP'09), Tianjin, China, 17-19 October 2009, Vol. 4, pp. 1993-1997.

[17] Y.-H. Yu, S. Kodagoda, and Q.P. Ha, "FPGA-Based Relative Distance Estimation for Indoor Robot Control Using Monocular Digital Camera", *Journal of Advanced Computational Intelligence & Intelligent Informatics*, to appear.

[18] J. Detrey and F. Dinechin, "Floating-Point Trigonometric Functions for FPGAs," *Proc. IEEE Intl. Conf. on Field Programmable Logic and Application*, Amsterdam, Hetherlands, 2007, pp. 29-34.

[19] P. P. Chu. *RTL Hardware Design Using VHDL*. John Wiley & Sons, Canada, 2006, pp. 460-468.