Sensors & Transducers, Vol. 25, Special Issue, December 2013, pp. 44-51

www.sensorsportal.com

Sensors & Transducers

© 2013 by IFSA

http://www.sensorsportal.com

Vision-based Parking Guidance with Adaptive Isometric Transformation

[*] Din-Chang Tseng, Ching-Chun Huang, and Tat-Wa Chao
The Institute of Computer Science and Information Engineering
National Central University, Jhongli, 32001, Taiwan
[*] Tel.: -886-3-4227151, fax: -886-3-4222681
[*] E-mail: tsengdc@ip.csie.ncu.edu.tw

Received: 16 September 2013 /Accepted: 15 October 2013 /Published: 23 December 2013

Abstract: A vision-based parking guidance system is proposed to help drivers parking their cars into parking space. The proposed system only relies on an embedded hardware and a wide-angle camera to capture images for analysis without steering sensor. This is a money-saved technique; moreover, it is suitable for used cars and after-market usage. In the system, input images are first transformed into top-view images by a homography transformation. Then corner feature points on two continuous images are extracted to match each other. The feature-point pairs are further pruned by a least-square error metrics. The remained pairs are then used to estimate vehicle motion parameters, where an isometric transformation model based on the Ackermann steering geometry is proposed to describe the vehicle motion. At last, the vehicle trajectory is estimated based on the vehicle motion parameters and the parking guidance lines are drawn according to the vehicle trajectory. The key characteristics of a parking guidance system are stability and precision; the proposed model is more proper than general considered models for image-based parking guidance system. *Copyright © 2013 IFSA.*

Keywords: Parking guidance, Parking assistance, Advanced safety vehicle, Ackermann geometry, Computer vision, Motion analysis.

1. Introduction

Parking a vehicle into a parking space or garage is an essential skill for drivers; however, it is still hard for someone. Moreover, to maneuver a vehicle into a parking space with limited dimension or with other vehicles or obstacles around the parking space is difficult for most drivers. Even a familiar driver has an unpleasant experience to park a vehicle into a small parallel parking space.

In these few decades, cameras and related embedded system are more and more cheap, and have been used for vehicle driving assistance, such as lane departure warning, forward collision warning, blind spot detection, *etc*. [1]. The mentioned parking problem can be solved by an around view monitor system [2-4] or a back guiding monitor and detection system, which are just equipped cameras and an embedded system.

Nissan Motor Co., Ltd. [2] has developed a parking assistance system named as "Around View Monitor". The system uses four wide-angle cameras mounted on the front, sides, and rear of the vehicle to capture images of the surrounding area to generate a bird's eye view of the vehicle with its surrounding scenery and a parking guiding view of the rear scenery as shown in Fig. 1.

Honda Motor Co., Ltd. [3] has provided a similar system named "Multi-view Camera System" with extra functions for tight driving support as shown in

Fig. 2. At the same time, Mitsubishi Motors Corp. [4] provided a similar system "Multi-around Monitor System". After two years, Fujitsu Semiconductor America, Inc. announced the commercial availability of a breakthrough 3D imaging technology that provides a complete 360-degree view of a vehicle's surrounding [5].

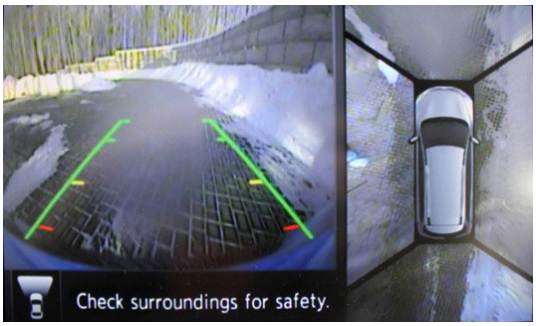


Fig. 1. Nissan Around view monitor system.

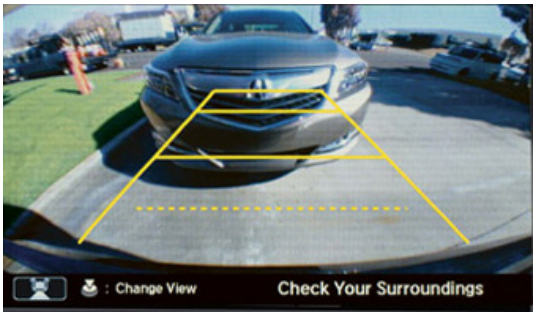


Fig. 2. Camera system.

In a general parking guidance system, a steering sensor is installed on the steering column to compute direction of front wheels and then provide the vehicle guiding path to the drivers. All above mentioned systems are just with such a configuration. The installation of steering sensor is a complicated task; it is also very inconvenient to install a steering sensor on a used car; thus the steering sensors are only set on new cars. Moreover, a steering sensor is expensive.

In this paper, we propose an image-based parking guidance (*IPG*) system to estimate the direction of front wheels without steering sensor. The hardware is just an *ARM*-based embedded system and a wide-angle camera. The camera is mounted on the rear of vehicle to capture sequential images of ground area just behind the vehicle. In the software development, off-line and on-line processes are sequentially constructed. The off-line process is used to calibrate 3D position of the camera to the ground coordinate system; then the on-line process is utilized to estimate the vehicle trajectory with respect to the ground coordinate system. At first, input images are first transformed into top-view images by a transformation matrix of homography. Then corner feature points on two continuous images are extracted to match each other. The feature-point pairs are further pruned by a least-square error metrics. The remained pairs are then used to estimate the motion parameters, where an isometric transformation model based on the Ackermann steering geometry [6] is proposed to describe the vehicle motion. A vehicle trajectory is then estimated based on the parameters. At last, a top view or rear view of parking guidance-lined images are provided to assist the driver. The proposed *IPG* system is shown in Fig. 3.

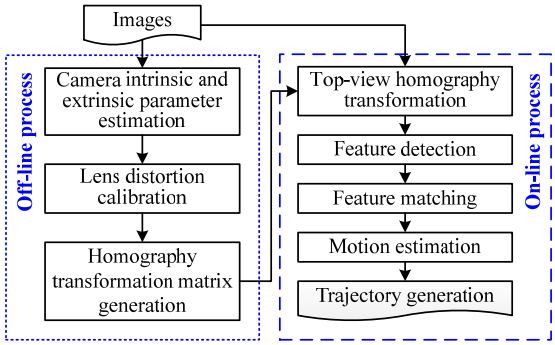


Fig. 3. The proposed image-based parking guiding system.

2. Camera Calibration

Camera calibration is a fundamental and important topic in computer vision and image processing. Many excellent works have been proposed [7]-[9]. Here we used the Zhang's camera calibration technique [10] and Devernay-Faugeras distortion calibration method [11] to estimate camera intrinsic, extrinsic, and distortion parameters.

In a perspective projection camera model, a 3D point $\boldsymbol{P} = [X\ Y\ Z\ 1]^{\mathrm{T}}$ and its 2D projection point $\boldsymbol{p} = [u\ v\ 1]^{\mathrm{T}}$ in the homogeneous coordinate system have the relation

$$s\,p = A\,[R\ t]\,P, \tag{1}$$

where s is the non-zero scale factor. $\boldsymbol{R}$ and $\boldsymbol{t}$ are 3×3 rotation matrix and 3×1 translation vector, respectively. $\boldsymbol{A}$ is the camera intrinsic parameters matrix:

$$\boldsymbol{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

where (u_o, v_o) is the camera optical axis center on the image coordinate system, α and β are the focal lengths in pixel unit for image u and v axes, respectively; γ is the skew parameter between photosensitive plane and optical axis of lens.

Based on the orthogonal restriction of coordinate axes, a complicated square-error formula is derived and then solved by the Levenberg-Marquardt algorithm [10]. In the Zhang's approach, the lens

distortion parameters were estimated based on the radial distortion model and the distortion was not completely resolved in the wide-angle images; thus Devernay-Faugeras' *FOV* distortion calibration method [11] was further utilized to improve the distortion calibration.

In the Devernay-Faugeras distortion model, the relation of a distortion image point $\boldsymbol{p}_d = [x_d\ y_d]^T$ and the undistorted image point $\boldsymbol{p}_u = [x_u\ y_u]^T$ is described as

$$r_d = R(r_u), \tag{3}$$

where $r_d = \sqrt{x_d^2 + y_d^2}$ and $r_u = \sqrt{x_u^2 + y_u^2}$. Based on the factor that the larger the incident angle between 3D point and optical axis, the larger the distance between image point and image center as shown in Fig. 4, Devernay and Faugeras [11] derived the field-of-view (*FOV*) model,

$$\boldsymbol{r}_d = \frac{1}{\omega}\tan^{-1}(2\,\boldsymbol{r}_u \tan\frac{\omega}{2}), \tag{4}$$

where ω is the distortion parameter associated to field of view; and the inverse function is

$$\boldsymbol{r}_u = \frac{\tan(\boldsymbol{r}_d\ \omega)}{2\tan\frac{\omega}{2}}. \tag{5}$$

One calibration example is shown in Fig. 5.

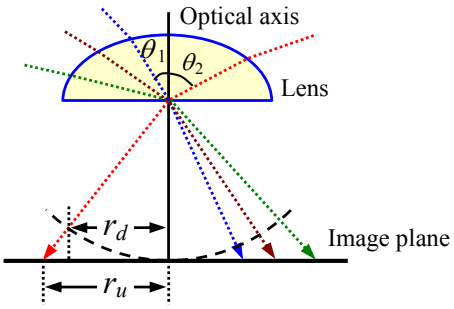


Fig. 4. Devernay-Faugeras *FOV* distortion model.

3. Top-view Transformation

There are several methods to generate top-view images. One of the simplest methods is using least-square calibration method to generate the top-view images. In the least-square method, we need a virtual image plane to present the desired top-view image. The idea is described as follows and shown in Fig. 6.

A calibration board with regular grid mesh is put in the view field of the rear camera. Assume a virtual camera is rightly above the calibration board facing down to the calibration board. It means that the virtual image plane is parallel to the calibration board; thus the grid mesh in the virtual image is completely like that on the calibration board. Consequently, we take the grid mesh on the capture image and calibration board to calibrate the relation between real and virtual cameras to generate the top-view images.

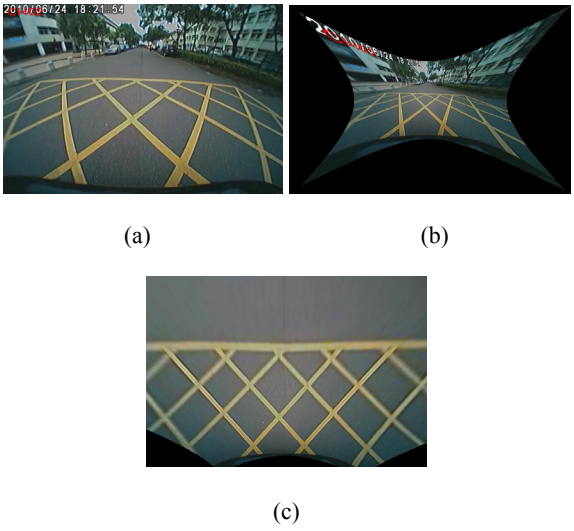


Fig. 5. An example of distortion correction. (a) Original image. (b) After distortion correction. (c) After top-view transformation.

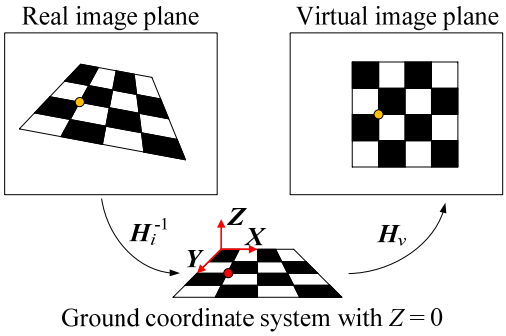


Fig. 6. The least-square calibration method.

Assume the coordinates of an image point p_i is $(u_i,\ v_i)$ and its corresponding point p_v on the calibration board is (x_i, y_i); then

$$s\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \boldsymbol{H}\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}, \tag{6}$$

where s is a scaling factor for adjusting the scale of the desired image. The formula is derived as

$$\begin{cases} u_i = \dfrac{h_{11}\,x_i + h_{12}\,y_i + h_{13}}{h_{31}\,x_i + h_{32}\,y_i + 1} \\ v_i = \dfrac{h_{21}\,x_i + h_{22}\,y_i + h_{23}}{h_{31}\,x_i + h_{32}\,y_i + 1} \end{cases}, \tag{7}$$

Then we can derive two linear equations from Eq.(7),

$$\begin{cases} h_{11}x_i + h_{12}y_i + h_{13} - h_{31}x_i u_i - h_{32}y_i u_i = u_i \\ h_{21}x_i + h_{22}y_i + h_{23} - h_{31}x_i v_i - h_{32}y_i v_i = v_i \end{cases}. \quad (8)$$

If there are n corresponding points on real and virtual images, we have $2n$ linear equations and then put these equations into a linear system as

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -y_1u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1v_1 & -y_1v_1 \\ & & & & \cdot & & & \\ & & & & \cdot & & & \\ & & & & \cdot & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nu_n & -y_nu_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_nv_n & -y_nv_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \cdot \\ \cdot \\ \cdot \\ u_n \\ v_n \end{bmatrix}. \quad (9)$$

The equation is simply written as

$$C_{2n\times 8}\, h_{8\times 1} = p_{2n\times 1}. \quad (10)$$

If $n \geq 4$, $\boldsymbol{x}$ can be calculated by the least square method

$$h = (C^{\mathrm{T}} C)^{-1} C^{\mathrm{T}} p. \quad (11)$$

$\boldsymbol{H}$ is then reconstructed from $\boldsymbol{h}$. The top-view image can then be inverse transform from the capture image by the homography matrix H. One example is shown in Fig. 5 (c).

The top-view transformation can also be completed by the previous camera calibration processes. Rewriting Eq.(1) in more detail,

$$s\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \boldsymbol{A}[\boldsymbol{R}\,\boldsymbol{t}]\begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} = \boldsymbol{A}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}\begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}. \quad (12)$$

If the 3D points are all on ground; that is, $Z_i = 0$, then the equation is simplified as

$$s\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \boldsymbol{A}\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix}\begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix} = \boldsymbol{H}_i\begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix}, \quad (13)$$

where $\boldsymbol{H}_i$ is just the homography transformation from ground plane to image plane. If we also construct the transformation matrix $\boldsymbol{H}_v$ for transforming from the ground plane to the virtual image plane; then $\boldsymbol{H}_v\,\boldsymbol{H}_i^{-1}$ is just the homogrpahy transformation matrix from real image plane to the virtual image plane.

4. Feature Matching

Matching two continuous images to acquire the motion parameters of the host vehicle, we need to extract feature points in advance. Rosten-Drummond *FAST* corner detector [12] was utilized to acquire corner feature points for matching.

Lowe [13] has proposed the scale-invariant feature transform (*SIFT*) for feature extraction and matching. *SIFT* is an excellent feature and method for application; however, it is very time consuming. In general, the vehicle speed is slow during parking and then the difference between two continuous images is small; thus a *SAD*-based matching method was alternatively used for feature-point matching.

For corner point (u, v) on image I_t matching a corner point (u'_k, v'_k) on image I_{t-1}, the sum of absolute difference (*SAD*) criterion is defined as

$$\min_{k=1,2,\ldots,K} SAD[(u,v),(u'_k,v'_k)] = \min_{k=1,2,\ldots,K} \sum_{i,j=-N/2}^{N/2} \left| f(u+i,v+j) - f(u'_k+i,v'_k+j) \right|, \quad (14)$$

where $f(u, v)$ is the gray level of pixel (u, v), K is the number of corner points on I_{t-1} and N is the radius of the window for computing a *SAD*. One example of corner-matching result is shown in Fig. 7.

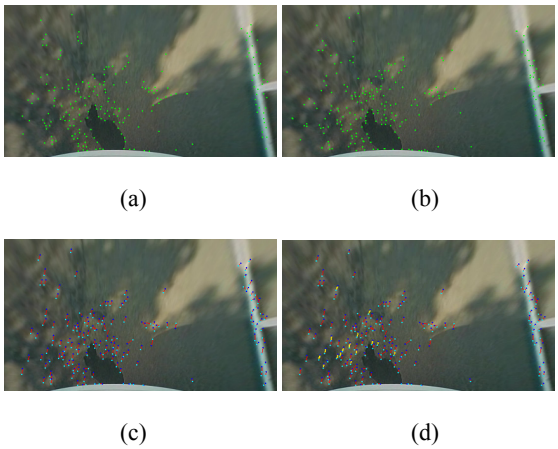


Fig. 7. Feature point matching. (a) The corner points on image I_t. (b) The corner points on image I_{t-1}. (b) The extracted point pairs. (c) The yellow lines are the credible point pairs.

5. Generation of Vehicle Trajectory

Ackermann geometry [6] described the motion trajectory of a four-wheel vehicle as shown in Fig. 8. In moving, all wheels have their axles arranged as radii of a circle with a common center point p. As the rear wheels are fixed, this center point must be on a line extended from the rear axle. Thus the center point can be found as the intersection of the rear-wheel and front-wheel axles.

We have no steering sensor, so we cannot get the directions of the front wheels directly. Here we use the movement of the vehicle to obtain the wheel direction based on the feature points on images.

We have designed two methods to generate vehicle trajectory. One is based on the concept of coordinate transformation and the other is based on the shift of centroid of the feature points.

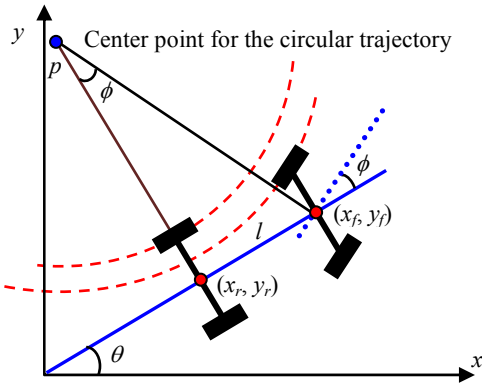


Fig. 8. Ackermann steering geometry and vehicular trajectory.

5.1. Coordinate Transformation Method

We analyze the geometric transformation between two continuous images based on the extracted feature point pairs as shown in Fig. 9. The isometric coordinate transformation model

$$\begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \left(\begin{bmatrix} u' \\ v' \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix} \right) \tag{15}$$

is used to simulate the Ackermann steering geometry to describe the vehicle motion. According to the Ackermann steering geometry, the vehicle is rotated with respect to a center point $p = (a, b)$. We use Eq.(15) to describe the vehicle motion from t-1 coordinate system (u', v') rotating θ angles to t coordinate system (u, v).

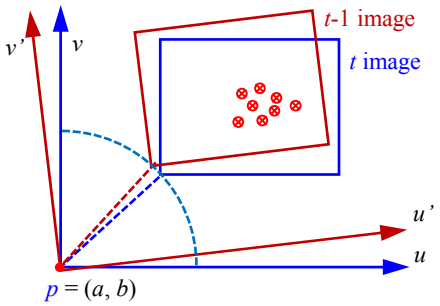


Fig. 9. The coordinate transformation model.

If we set $c = \cos\theta$ and $d = \sin\theta$, then Eq.(15) can be rewritten as two linear equations

$$\begin{cases} u = c\,(u'+a) - d\,(v'+b) - a \\ v = d\,(u'+a) + c\,(v'+b) - b \end{cases}, \tag{16}$$

and then becomes

$$\begin{cases} u = c\,u' - d\,v' - a + c\,a - d\,b \\ v = d\,u' + c\,v' - b + d\,a + c\,b \end{cases}. \tag{17}$$

Let

$$\begin{cases} e = (c-1)\,a - d\,b \\ f = d\,a + (c-1)\,b \end{cases}, \tag{18}$$

then Eq.(17) becomes two linear equations of four unknowns c, d, e, and f,

$$\begin{cases} u = c\,u' - d\,v' + e \\ v = d\,u' + c\,v' + f \end{cases}. \tag{19}$$

If there are n corresponding points on the two continuous images, we can get $2n$ linear equations. Integrating these equations to form a linear system

$$\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \end{bmatrix}_{2n\times 1} = \begin{bmatrix} u'_1 & -v'_1 & 1 & 0 \\ v'_1 & u'_1 & 0 & 1 \\ u'_2 & -v'_2 & 1 & 0 \\ v'_2 & u'_2 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ u'_n & -v'_n & 1 & 0 \\ v'_n & u'_n & 0 & 1 \end{bmatrix}_{2n\times 4} \begin{bmatrix} c \\ d \\ e \\ f \end{bmatrix}_{4\times 1}. \tag{20}$$

The equation is simply written as

$$b = A\,x, \tag{21}$$

then the unknowns (c, d, e, f) can be estimated by the least-square estimation method,

$$x = (A^{\mathrm{T}} A)^{-1} A^{\mathrm{T}} b. \tag{22}$$

At last, the center point (a, b) can be found by Eq.(18)

$$\begin{bmatrix} c-1 & -d \\ d & c-1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} e \\ f \end{bmatrix}; \tag{23}$$

$$\begin{cases} a = \dfrac{e\,(c-1) + d\,f}{(c-1)^2 + d^2} \\ b = \dfrac{(c-1)\,f - d\,e}{(c-1)^2 + d^2} \end{cases}. \tag{24}$$

Since $\cos^2\theta + \sin^2\theta = 1$; that is, $c^2 + d^2 = 1$; thus

$$\begin{cases} a = \dfrac{e\,(c-1) - d\,f}{2-2c} \\ b = \dfrac{(c-1)\,f + d\,e}{2-2c} \end{cases} \tag{25}$$

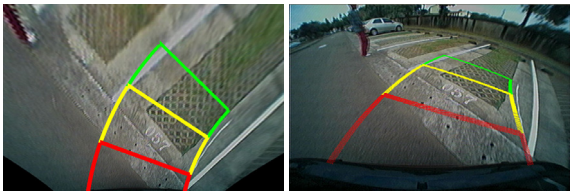
and $\theta = \tan^{-1}(d/c)$.

Some point pairs may involve more error; thus these point pairs should be pruned out. We utilize the square error to prune the point pairs which have more error. Substituting a point pair $[(u_i, v_i)\ (u'_i, v'_i)]$ into Eq.(19) to compute the square error

$$\varepsilon_i = [u_i - c\,u'_i + d\,v'_i - e)]^2 + [v_i - d\,u'_i - c\,v'_i - f)]^2. \tag{26}$$

If the error is greater than a pre-defined threshold value, the point pair is discarded. After that, all remained point pairs are used to re-compute the least-square estimation of transformation parameters again. At last, the parameters a, b, and θ are used to generate the vehicular trajectory.

With the coordinate transformation method, the vehicular trajectory is generated and then the parking guiding lines are drawn on top-view and original images as examples shown in Fig. 10.



(a) (b)

Fig. 10. The parking guiding lines. (a) drawn on the top-view image. (b) drawn on the original image

5.2. Centroid Shift Method

This second method needs no feature-point matching and then is time saved. The centroid of all extracted feature points in one image is used to define the location of the vehicle in the world coordinate system. The centroid is regarded as independence to rotation; thus two centroids of feature points in two continuous images are used to find the move direction of the front wheels as shown in Fig. 11.

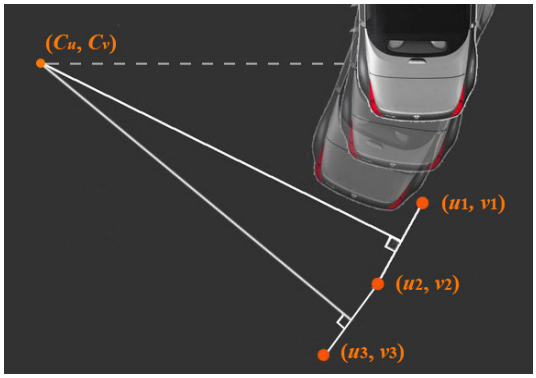


Fig. 11. The vehicular trajectory model on world coordinate system.

Assume the centroid coordinates of feature points in images I_{t-1} and I_t are (u_1, v_1) and (u_2, v_2), respectively. The trajectory arc passes through points (u_1, v_1) and (u_2, v_2); the line between (u_1, v_1) and (u_2, v_2) must be a chord of the circle trajectory. The perpendicular line of the chord must pass through the center point (c_u, c_v) of the circle trajectory; so the equation of the perpendicular line (radius of the circle) can be defined as

$$v - (\frac{v_1 + v_2}{2}) = e\,(u - (\frac{u_1 + u_2}{2})), \quad (27)$$

where $e = -(u_2 - u_1)/(v_2 - v_1)$ is the slope of the perpendicular line.

We have stated that the center point is located on the extended line of the rear axle. If current image is I_t, the rear axle is parallel to the u-axis of image. Assume the equation of the rear wheel axle line is $v = v_0$, where v_0 is a constant and was previously calibrated. The center point coordinates are just the intersection of two axles. The coordinate of c_u can be obtained by substituting $v_c = v_0$ into Eq.(27),

$$c_u = \frac{u_1 + u_2}{2} + \frac{1}{e}(v_c - \frac{v_1 + v_2}{2}). \quad (28)$$

6. Experiments

The proposed methods were implemented with C++ language, tested on a PC-based platform with Intel® Core2 Duo 2.66 *GHz CPU*, and run on an embedded system with 1 *GHz ARM* Cortex A8 *CPU*. A low-cost wide-angle camera with a view field of 136° angle in horizontal and 115° angle in vertical was used to capture images.

In the current implementation of generating vehicular trajectory, although the centroid shift method is faster than the isometric coordinate transformation method; the isometric coordinate transformation method generated less error than the centroid shift method; thus the isometric coordinate transformation method was adopted in the following experiments and analysis. Both perpendicular parking and parallel parking to parking spaces were demonstrated as examples shown in Figs. 12 and 13. The guiding lines are drawn according to the transformation parameters in real time. Run on the embedded system with 20 frames per second. The top-view parking guidance images can also been alternatively provided to the drivers.

Fig. 12. A sequence images of perpendicular parking.

We totally generated six vehicle trajectories with different steering angles to analyze the accuracy of

the guiding lines. The steering angles include 0, 90, 180, 270, 360, and 450 degrees. The average and standard deviation of the deviation distances on each trajectory are given in Table 1. The deviation errors are small in small steering angles, but the errors are growing during the steering angles are increased. The proposed isometric coordinate transformation is theoretically proper to the geometric transformation of vehicle motion; however the (rotation) center point is very sensitive to the extracted point pairs. Different sets of point pairs in multiple continuous images may result in vibrated center points and then generate guiding lines with larger errors.

Fig. 13. A sequence images of parallel parking.

Table 1. Accuracy of the generated vehicle trajectories.

	0°	90°	180°	270°	360°	450°
Average	2.03	3.28	4.16	5.73	7.65	8.22
Standard deviation	1.52	2.31	3.27	3.84	4.93	6.36
Maximum	6.54	12.64	16.39	20.71	22.44	30.34

7. Conclusions

In this paper, a vision-based parking guidance (*IPG*) system is proposed to help drivers parking their cars into parking space. The proposed system only uses an embedded hardware and a wide-angle camera to capture images for vehicle trajectory generation; the proposed system needs no steering sensor; it is a money-saved technique; moreover, it is suitable for used cars and after-market usage.

The proposed system only relies on image analysis; it may results in larger error in the guiding lines. The isometric transformation is theoretically proper to the geometric transformation of vehicle motion; however the (rotation) center point is very sensitive to the extracted point pairs. The trajectory is vibrated during driving backward.

In the future, a motion analysis method will be proposed to estimate the geometric transformation. It is directly using optical flow to estimate motion vectors for trajectory generation without feature matching. The least-square-version Lucas-Kanade optical flow method [14, 15] can used to estimate the motion vectors. On the other hand, a stabilization method based on the learning principle will also be provided to stabilize the trajectory. We hope that the stability of parking guiding lines generated by the proposed method is the same as that of using steering sensor. Moreover, the obstacle detection [16] should be added to enhance the system benefit.

Acknowledgements

This work was supported in part by the National Science Council, Taiwan under the grant of the research project NSC 100-2221-E-008-115-MY3.

References

[1]. R. Bishop, Intelligent Vehicle Technology and Trends, *Artech House*, Boston, Massachusetts, 2005.

[2]. Around View Monitor, *Nissan,* http://www.nissan-global.com/en/technology/overview/avm.html

[3]. Multi-View Camera System, *Honda,* http://world.honda.com/news/2008/4080918 Multi-View-Camera-System/

[4]. K. Ueminami, K. Hayase, and E. Sato, Multi-around monitor system, *Mitsubishi Motors Technical Review*, No. 19, 2007, pp. 55-58.

[5]. 360-Degree Wrap-around Video Imaging Technology, *Fujitsu,* http://www.fujitsu.com/us/news/pr/ fma_20101019-02.html

[6]. D. King-Hele, Erasmus Darwin's improved design for steering carriages-and cars, *Notes and Records of the Royal Society of London*, Vol. 56, No. 1, January 2002, pp. 41-62.

[7]. R. Y. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses, *IEEE Jour. Robotics and Automation*, Vol. 3, No. 4, 1987, pp. 323-344.

[8]. J. Weng, P. Cohen, and M. Herniou, Camera calibration with distortion models and accuracy evaluation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 14, No. 10, 1992, pp. 965-980.

[9]. S. Li and Y. Hai, Easy calibration of a blind-spot-free fisheye camera system using a scene of a parking space, *IEEE Trans. on Intelligent Transportation Systems*, Vol. 12, No. 1, 2011, pp. 232-242.

[10]. Z. Zhang, A flexible new technique for camera calibration, *IEEE Trans. on Pattern Analysis and Machine Intell.,* Vol. 22, No. 11, 2000, pp. 1330-1334.

[11]. F. Devernay and O. Faugeras, Straight lines have to be straight, *Machine Vision and Applications,* Vol. 13, No. 1, 2001, pp. 14-24.

[12]. E. Rosten and T. Drummond, Machine learning for high-speed corner detection, *Lecture Notes in Computer Science* (*ECCV*" 06), Vol. 3951, 2006, pp. 430-443.

[13]. D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, Vol. 60, Nov. 2004, pp. 91-110.

[14]. B. D. Lucas and T. Kanade, An iterative image registration technique with an application to stereo vision, in *Proceedings of the Int. Joint Conf. on*

Artificial Intelligence, Vancouver, Canada, August 24-28, 1981, pp. 674-679.

[15]. J. Barron, D. Fleet, and S. Beauchemin, Performance of optical flow techniques, *International Journal of Computer Vision*, Vol. 12, No. 1, 1994, pp. 43-77.

[16]. I. Ullah, F. Ullah, Q. Ullah, and S. Shin, Sensor-based robotic model for vehicle accident avoidance, *Journal of Computational Intelligence and Electronic Systems*, Vol. 1, No. 1, 2012, pp. 57-62.

2013 Copyright ©, International Frequency Sensor Association (IFSA). All rights reserved. (http://www.sensorsportal.com)