# Extending Driving Vision Based on Image Mosaic Technique

Deng Chen[1,*], Yanduo Zhang[1], Wei Wei[1], Xiaolin Li[1], Huabing Zhou[1], Rui Zhu[1], Tao Lu[1], Xun Li[1], Haijiao Xu[2] and Li Peng[3]

[1] Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, China
[2] Department of Computer&Science, Guangdong University of Education, Guangzhou, China
[3] Teaching Guidance Center, Hubei Radio & TV University, Wuhan, China
Corresponding Email: dchen@wit.edu.cn

**Abstract.** Car cameras have been used extensively to assist driving by make driving visible. However, due to the limitation of the Angle of View (AoV), the dead zone still exists, which is a primary origin of car accidents. In this paper, we introduce a system to extend the vision of drivers to 360 degrees. Our system consists of four wide-angle cameras, which are mounted at different sides of a car. Although the AoV of each camera is within 180 degrees, relying on the image mosaic technique, our system can seamlessly integrate 4-channel videos into a panorama video. The panorama video enable drivers to observe everywhere around a car as far as three meters from a top view. We performed experiments in a laboratory environment. Preliminary results show that our system can eliminate vision dead zone completely. Additionally, the real-time performance of our system can satisfy requirements for practical use.

## 1 Introduction

In order to reduce car accidents, many electronic devices have been introduced to assist driving, such as parking sensors and Rear-View Car Cameras (RVCCs) [1]. RVCCs facilitate driving enormously by making it visible. However, the Angle of View (AoV) of existing RVCC is less than 180 degrees, that is, there exists dead zone outside our view, which is a primary origin of traffic accidents. To resolve the problem, we design a vehicle panorama system, which can extend the vision of drivers to 360 degrees around a car.

Our system comprises four wide-angle cameras, each of which has an AoV near to 180 degrees. The cameras are mounted at different sides around a car. Obviously, the cameras can capture 4-channel videos around a car. Via switching among the videos, drivers can observe everywhere around a car. However, switching videos may cause distraction from driving. Additionally, the experience is bad to observe four distinct videos, especially when videos overlapped with each other. In order to mitigate the problem, we seamlessly integrate the 4-channel videos into a panorama video. The image processing algorithms used by our system are camera calibration, image distortion correction and image mosaic. The output of our system includes a panorama video and a side video. The panorama video enables drivers to observe the environment around a car as far as three meters from a top view. The side video can be switched among the four cameras automatically. In order to investigate the effect of our system, we conduct experiments based on a mimical car and good results are achieved.

The contributions of this paper are:

• Hardware design of a driving assist system based on DM6437, which can extend driving vision to 360 degrees around a car;

• Image processing algorithms used by our system;

• The investigation of the effect of our system.

The rest of this paper is organized as follows: Section 2 presents the general view of our system. Section 3 introduces our image mosaic technique. Section 4 reports our preliminary experimental results. Section 5 concludes this paper.

## 2 General view of our system

The hardware of our system mainly consists of the following components: CCD Cameras, TVP5158, DM6437, DDR2 RAM, NOR Flash, Video Output Connector and CH376. Its functioning is illustrated in Fig. **1**. As we can see, it captures video images through four wide-angle cameras mounted at different sides of a car. The video decoder

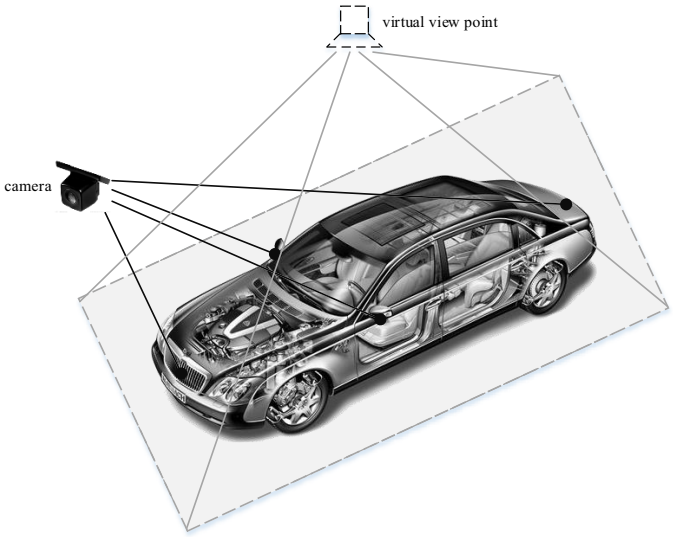


**Fig. 1.** The functioning of our system.

TVP5158 digitizes and decodes all popular base-band analog video formats into digital video output [2]. The main processor DM6437 [3-7] takes the digital videos as input and processes images based on OpenCV [8]. The storage of our system comprises a 16M NOR Flash and a 128M DDR2 RAM. The former is used to save programs and configuration information. The latter is used to cache videos. Additionally, relying on the file management controller CH376, our system can upgrade software through USB flash drives or SD cards.

Based on the above design, our system can provide drivers with a panorama image around a car from a virtual view point as illustrated in Fig. **1**. The virtual view point is above the centre of a car. Its height can be assign a value between zero and ten meters. By raising the virtual view point, we can broaden the panorama view to a maximum extent of three meters around a car.

## 3 Image processing

Our system processes images based on the open sourced computer vision library OpenCV. In this section, we elaborate the key image processing algorithms used by our system, inclusive of camera calibration [9,10], image distortion correction [11, 12] and image mosaic [13, 14].

### 3.1 Camera calibration

Camera calibration is a necessary step for many image processing tasks, such as 3D computer vision and image distortion correction. Through camera calibration, we can achieve the following metric information about a camera from 2D images:

- The extrinsic parameters $(\mathbf{R}, \mathbf{t})$ of a camera, where $\mathbf{R}$ is a 3 × 3 rotation matrix, $\mathbf{t}$ is a 3D translation vector;
- The intrinsic matrix $\mathbf{A}$ of a camera, which is given by

$$\mathbf{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

with $(u_0,\ v_0)$ the coordinates of the principle point, $\alpha$ and $\beta$ the scale factors in image $u$ and $v$ axes, and $\gamma$ the parameter describing the skewness of the two image axes;

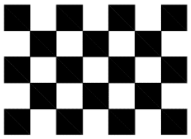

**Fig. 2.** The model plane used for calibration.

- The coefficients of radial distortion $k_1$ and $k_2$;

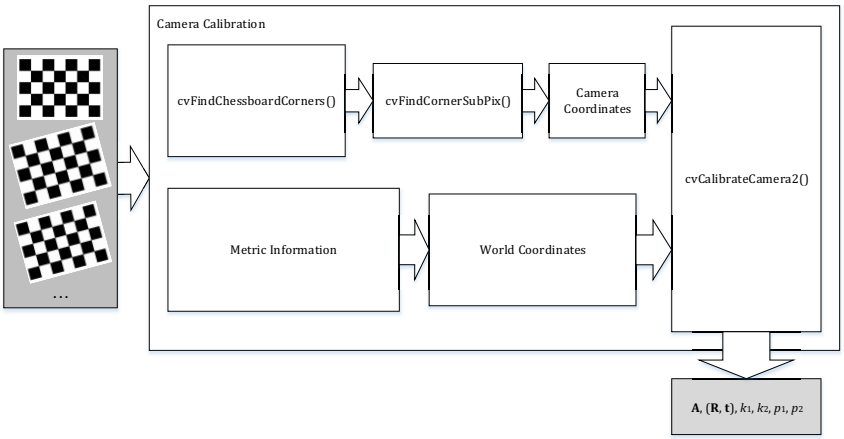


**Fig. 3.** The working principle for camera calibration.

• The coefficients of tangential distortion $p_1$ and $p_2$.

The extrinsic parameters and intrinsic matrix relate the world coordinate system to the camera coordinate system. Let $\tilde{\mathrm{m}}$ and $\tilde{\mathrm{M}}$ be a 2D and a 3D point respectively (homogeneous coordinates). The relationship between a 3D point $\tilde{\mathrm{M}}$ and its image projection $\tilde{\mathrm{m}}$ is given by

$$s\tilde{\mathrm{m}} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}]\tilde{\mathrm{M}},$$

where $s$ is an arbitrary scale factor. The distortion coefficients are essential for the image distortion correction.

We calibrate cameras using the method proposed by Zhang [10], which only requires the camera to observe a planar pattern shown at a few different orientations. As shown in Fig. **2**, our system utilizes a chessboard for calibration. It contains a pattern of 7 × 5 squares, so there are 24 internal corners (i.e., points where the black squares touch each other.) The size of the chessboard is 12.9 × 9.2 centimeters and that of each square is 1.9 × 1.9 centimeters. We print the chessboard with a high-quality printer and then put it on a plastic board. Before calibrating a camera, we hold the plastic board by hand and place it under the camera as far as its image fills in the whole screen. Our system takes eight gray images of the chessboard as input of the calibration algorithm (the gray image can be obtained by extracting the Y channel of an image in format of YCbCr-422.) Since the hand will inevitably shake, the eight images must be under different orientations.

The working principle of the calibration algorithm is shown in Fig. **3**, which mainly consists of three OpenCV functions, i.e., cvFindChessboardCorners(), cvFindCornerSubPix() and cvCalibrateCamera2(). The first function attempts to determine whether the input image is a view of the chessboard pattern and locate the internal chessboard corners. A non-zero value is returned if all of the corners are found and ordered. Otherwise, it returns zero. What should be noted is that the coordinates detected by cvFindChessboardCorners() are approximate. To determine their position more accurately, we should subsequently call the function cvFindCornerSubPix() which can refine the results obtained by cvFindChessboardCorners() and achieve the sub-pixel location of corners. After that, we compute the world coordinates of the corners based on the metric information of the chessboard. Finally, we take the camera coordinates achieved by cvFindCornerSubPix() and the computed world coordinates as input and call the function cvCalibrateCamera2() to estimate the intrinsic camera parameters, distortion coefficients and extrinsic parameters for each of the chessboard images.

### 3.2 Image distortion correction

As shown in Fig. **4**, serious distortion exists in the images captured by our system. To facilitate image mosaic, we should correct them. We perform the image distortion correction based on the OpenCV function cvUndistort2(). Let's assume that $G'$ is the corrected image of $G$. cvUndistort2() corrects image $G$ by mapping each pixel of $G'$ to a pixel in $G$. The mapping is calculated based on the intrinsic matrix **A** and distortion coefficients $k_1$, $k_2$, $p_1$ and $p_2$, which are achieved by the above camera calibration. The corrected image output by cvUndistort2() has the same size and type as the input image. What should be noted is that the function cvUndistort2() only works with gray images.

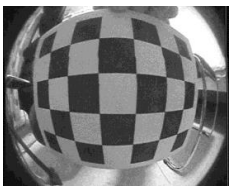

**Fig. 4.** An example image captured by our system.

### 3.3 Image mosaic

Our system performs the task of image mosaic based on a two-phase approach. The first phase is carried out only once, in which we calculate and save the image projection parameters into the Flash. The second phase is executed in each frame, which integrates the 4-channel images into a whole based on the projection parameters achieved in the first phase. Below we will elaborate our approach in detail.

As we can see from Fig. **5** (a), Ow-XwYwZw is the world coordinate system. Its origin is located on the ground, at the center of the car. We assume that there exists a virtual camera, which is $h$ meters high above the center of the car. Its camera coordinate system is Oc-XcYcZc. The goal of our image mosaic technique is to project images captured by four cameras into the image $I_v$, which is output by the virtual camera. Let $I_1$, $I_2$, $I_3$, $I_4$ be the corrected images captured by the front, rear, left and right cameras respectively. In order to project these images to $I_v$, we should achieve the homography matrixes $\mathbf{H}_i$, $i = 1...4$, which are between $I_i$ and $I_v$, respectively. As shown in Fig. **5** (a), we compute the homography matrixes based on four chessboards, which are the same as that used for camera calibration, except that they have different sizes. The size of the chessboards used for image mosaic depends on the installation height of cameras. To prepare for image mosaic, we lay all chessboards on the ground, around the car. The front and rear ones are in the middle of the car. The left and right ones are $l_1$ and $l_2$ meters off the front of the car. What should be noted is that, the chessboards should be placed in a position where their images fill in the screen as much as possible. Obviously, based on the above metric information and the size of the car, we can calculate the world coordinates of all chessboard corners conveniently. Assuming $P_w$ to be the set of corner points (world coordinates) of the front chessboard, our method of computing the homography matrix proceeds as follows:

1) Projecting Pw to the set of image points Pv from the perspective of the virtual camera. This task can be accomplished with the OpenCV function cvProjectPoints2(), which takes Pw and the parameters of the virtual camera as input.
2) Finding the set of corner points P1 from image I1 captured by the front camera. According to the introduction of camera calibration, this task can be carried out by using OpenCV functions cvFindChessboardCorners() and cvFindCornerSubPix().
3) Calculating the homography matrix H1 based on Pv and P1 achieved in step 1) and 2) respectively. We perform this task leveraging the OpenCV function cvFindHomography().
4) Computing the inverse matrix $\boldsymbol{H}_1^{-1}$ of H1 by using the OpenCV function cvInvert().

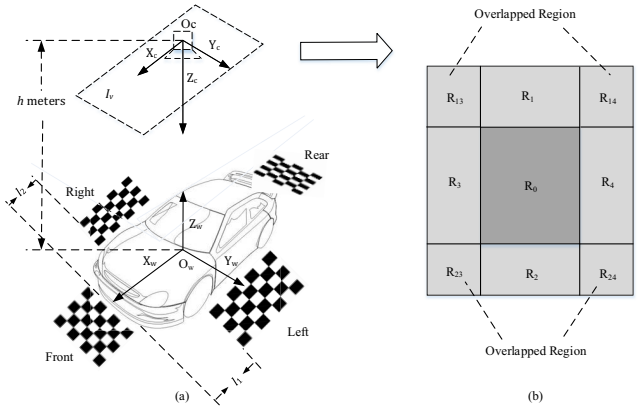


**Fig. 5.** The functioning of our image mosaic technique.

After that, we construct image $I_v$ based on the following equation:

$$\mathrm{clr}(I_v, P) = \begin{cases} \mathrm{clr}(I_{car}, SP) & if\ P \in R_0 \\ \mathrm{clr}(I_1, \mathrm{H}_1^{-1}P) & if\ P \in R_1 \\ \mathrm{clr}(I_2, \mathrm{H}_2^{-1}P) & if\ P \in R_2 \\ \mathrm{clr}(I_3, \mathrm{H}_3^{-1}P) & if\ P \in R_3 \\ \mathrm{clr}(I_4, \mathrm{H}_4^{-1}P) & if\ P \in R_4 \\ \mathrm{mrg}\big(\mathrm{clr}(I_1, \mathrm{H}_1^{-1}P), \mathrm{clr}(I_3, \mathrm{H}_3^{-1}P)\big) & if\ P \in R_{13} \\ \mathrm{mrg}\big(\mathrm{clr}(I_1, \mathrm{H}_1^{-1}P), \mathrm{clr}(I_4, \mathrm{H}_4^{-1}P)\big) & if\ P \in R_{14} \\ \mathrm{mrg}\big(\mathrm{clr}(I_2, \mathrm{H}_2^{-1}P), \mathrm{clr}(I_3, \mathrm{H}_3^{-1}P)\big) & if\ P \in R_{23} \\ \mathrm{mrg}\big(\mathrm{clr}(I_2, \mathrm{H}_2^{-1}P), \mathrm{clr}(I_4, \mathrm{H}_4^{-1}P)\big) & if\ P \in R_{24} \end{cases} \quad (1)$$

where $P$ is a point (homogeneous coordinates) in image $I_v$. Given an image $I$ and a point $p$ in $I$, $\mathrm{clr}(I, p)$ represents the color of pixel $p$ in $I$. $I_{car}$ denotes a car image, which is in the format of RGB565. $S$ is a 3D vector $(sx, sy, 1)$, where $sx$ and $sy$ are the horizontal and vertical scale factors between $I_{car}$ and $I_v$ respectively. $R_0$~$R_4$, $R_{13}$, $R_{14}$, $R_{23}$ and $R_{24}$ are different regions of image $I_v$, which are shown in Fig. **5** (b). According to the composition of pixels, we can divide these regions into three categories: 1) Region $R_0$ is the projection of the car, which will be filled with a car image. Its size can be determined according to the projection of car corners. 2) Regions $R_1$, $R_2$, $R_3$ and $R_4$ comprise pixels coming from $I_1$, $I_2$, $I_3$ and $I_4$, respectively. 3) Regions $R_{13}$, $R_{14}$, $R_{23}$ and $R_{24}$ are overlapped regions, that is, a pixel of the regions may be the fusion of two pixels originating from different cameras. For example, a pixel of $R_{13}$ may be a mixture of two pixels coming from images $I_1$ and $I_3$ respectively. For an overlapped pixel, we compute its color based on the mrg function, which takes two source pixels as input and outputs the mixed pixel. The mrg function is crucial for our system. Taking a pixel $(x, y) \in R_{13}$ for example, we utilize the following mrg function in this work:

$$\mathrm{mrg}(x, y) = w_1(x_1, y_1)I_1(x_1, y_1) + w_3(x_3, y_3)I_3(x_3, y_3) \quad (2)$$

where $w_i(x_i, y_i)$, $i \in \{1,3\}$ is the weight function. Its form is given as follows:

$$w_i(x_i, y_i) = \frac{k_i(x_i, y_i)}{\sum_i k_i(x_i, y_i)}$$

$$k_i(x_i, y_i) = \left(1 - \left|\frac{x_i}{width(I_i)} - \frac{1}{2}\right|\right) \times \left(1 - \left|\frac{y_i}{height(I_i)} - \frac{1}{2}\right|\right)$$

Additionally, we have

$$(x_i \quad y_i \quad 1)^T = \mathrm{H}_i^{-1}(x \quad y \quad 1)^T$$

According to [15], the image mosaic technique based on the above weight function should work well.

In order to improve the real-time performance of our system, for each pixel $(x, y)$ in image $I_v$, we save the following information into the Flash in the first phase: 1) pixel $(x_1, y_1)$ in the first source image; 2) pixel $(x_2, y_2)$ in the second source image; 3) the weight of the first source pixel; and 4) the weight of the second source pixel. If a pixel has only one source pixel (e.g. pixels in regions $R_0$~$R_4$), we invalidate the other source pixel by setting its weight to zero. Based on the results of the first phase, we can generate the panorama image $I_v$ from (2) directly in each frame. Our method is efficient, because the first phase is performed only once. In each frame, we carry out only two multiplications and one addition for each pixel.

## 4 Experiments

In order to investigate the effect of our system, we performed experiments based on a mimical car, which is a cardboard box of the size of 1.4 × 1.2 meters. The chessboards used for image mosaic have the same size of an A4 paper. We mounted the cameras on the four sides of the cardboard box and launched our system. The effect of our system is shown in Fig. **6**. As we can see, the panorama video is displayed on the left side, from which we can observe everywhere around a car from a top view. The side video is displayed on the right side, which can be switched automatically among the four cameras.

We also investigated the real-time performance of our system and found that our system achieved 25 frames per second on average, which can satisfy the requirements for practical use.

**Fig. 6.** The effect of our system.

## 5 Conclusions

In this paper, we gave the design of a vehicle panorama system. Our system leverages the techniques of camera calibration, image distortion correction and image mosaic to provide drivers with a panorama view of a car. The panorama view is able to extend the vision of drivers to 360 degrees, which provides a strong assurance for driving cars safely. In this work, we introduced design and implementation details of our system. We believe it is a good reference for the development of similar systems.

## Acknowledgements

This work was supported by the Youths Science Foundation of Wuhan Institute of Technology (No. k201622), Surveying and Mapping Geographic Information Public Welfare Scientific Research Special Industry (No. 201412014), National Natural Science Foundation of China (No. 41501505, 61502355 and 61502354), Nature Science Foundation of Hubei Province (No. 2014CFB779) and the Key Program of Higher Education Institutions of Henan Province (No. 17A520040).

## References

1. J. Ju, G. Han, M. Lin, et al., Design of a vehicle panorama reversing system based on embedded system, in *Chinese Automation Congress (CAC)*, Wuhan, China, pp. 1528-1532 (2015)
2. Texas Instruments, Data Manual for TVP5158, TVP5157 and TVP5156 (2011)
3. L. Kang, Z. Zhao, G. Xie, The hardware design of dual-mode wireless video surveillance system based on DM6437, in *2nd International Conference on Networks Security, Wireless Communications and Trusted Computing*, pp. 546-549 (2010)
4. D. Chen, Y. Zhang, W. Wei, et al., Design of a panorama parking system based on DM6437, in *Proceedings of the 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics* (to be published)
5. B. Li, W. Zhang, Z. Liu, et al., Development and implement of the IP camera, in *2011 International Conference on Mechatronic Science, Electric Engineering and Computer*, Jilin, China, pp. 1961-1964 (2011)
6. Texas Instruments, User's Guide for TMS320DM643x DMP Video Processing Back End (VPBE) (2007)
7. Texas Instruments, User's Guide for TMS320DM643x DMP Video Processing Front End (VPFE) (2007)
8. OpenCV, http://opencv.org (2016)
9. Q. Chen, H. Wu, S. Higashino, et al, Camera calibration by recovering projected centers of circle pairs, in *ACM SIGGRAPH 2016 Posters*, Anaheim, California, pp. 1-2 (2016)
10. Z. Zhang, A flexible new technique for camera calibration, IEEE Transactions on Pattern Analysis and Machine Intelligence **22**, 11, pp. 1330-1334 (2000)
11. M.S. Brown, D. Tsoi, Correcting common distortions in camera-imaged library materials, in *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, Houston, Texas, pp. 367-368 (2003)
12. S. Lee, J. Lee, J. Choi, Lens distortion correction using a checkerboard pattern, in *Proceedings of the 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, Singapore, pp. 1-2 (2008)
13. X. Shao, C. Xu, J.H. Lim, Image mosaics base on homogeneous coordinates, in *2002 Pan-Sydney Workshop on Visualisation*, Sydney, Australia, pp. 93-96 (2002)
14. R. Szeliski, H. Shum, Creating full view panoramic image mosaics and environment maps, in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 251-258 (1997)
15. Q. Ruan, Shi shi shi pin pin jie xi tong guan jian ji shu yan jiu, M. Chs thesis, Huazhong University of Science and Technology, Wuhan, China (2011)