Image Recognition Technology and Optimized Control Algorithm of the Self-tracing Smart Car Based on CMOS Camera Sensor

Zihui Wang[1], Yunyue Ye[2]
Department of Linear motor and drives, Zhejiang University of Electrical Engineering, China
[1]Email: wzh2718@126.com [2]yeyunyue@yahoo.com.cn

***Abstract* — As an intellegent system, the self-tracing smart car contains a core of MCS12 micro control unit (MCU) of Freescale. Acquiring useful information provided by a CMOS camera, the system can recognize the pathway and give proper parameters, as well as choosing optimized control algorithm for image acquisition, noise filtering, shape reforming, PID or fuzzy logical control and memory methods for precise control of the servo and driver motors. Experiments demonstrate that this car has a good ability to regulate direction along pathway and to response acceleration. So it can successfully complete the race as fast as possible in the diverse competition.**

I. INTRODUCTION

The self-tracing smart car usually uses photoelectric cells or cameras as sensors to detect pathway information. The camera sensor generally has the advantage of farther prospective distance and wider visual range than photoelectric sensors, which is more effective to anticipate raceway conditions and then make quick running or cut turns. However, camera sensor system costs more memory storage space and longer processing time. Therefore, good algorithms and effective storage methods are highly needed.

The actuating mechanism of the car consists of a DC motor and a servo motor, supplied by 7.2V and 6V DC power. Two channels of PWM signals at TTL voltage level are output from S12 Micro Control Unit (MCU) to control the speed and orientation. One PWM signal goes into MC33886 (Driver IC) and then drive the back DC motor, while the other goes directly into front servo motor and make a precise rotation. Both actions are related to the pulse width of the PWM signals. Therefore, the PWM control signals are very important to manipulations of the car. Furthermore, good control of PWM signals must depends on a precise real-time detection and optimized control algorithms to choose appropriate parameters of the raceway information.

In this paper, a matrix CMOS camera is used as eyes of the smart car. Considering the problem of sampling frequency limit (50Hz) of the CMOS camera, this algorithm achieves up to 80 sampling points per line per frame without overclocking the MCU—at the normal frequency of 24MHz. Therefore, the image resolution is highly improved, which makes it easier to recognize trace and more accurate in checking start point. The recognition technology involves image acquisition, noise filtering, shape reforming, PID or fuzzy logical control of turning and memory methods of pathway.

II. IMAGE ACQUISITION TECHNOLOGY

The objective of this smart car is to follow a black line painted on the white floor by using a simple camera to sense how the road curves and shifts. The self-tracing smart car usually uses photoelectric cells and cameras as sensors to detect raceway information. Generally, we use photoelectric cell to detect information because of its prevalent usage. The photoelectric cell emits infrared and receives different voltage signals representing the grey value of the ground. About 7 to 13 photoelectric cells should be mounted side by side 5 to 10cm ahead of the car, so that they can recognize the color of the road and analyze the raceway information. However, photoelectric cell can only be installed close to the ground because of its very little forward-looking distance. In addition, one photoelectric cell can only detect one point of the image, while only less than 16 sensors are allowed by the competition rule. That means we can get very little information from it. Those disadvantages create unsolvable difficulties for the smart car to run and turn at a high speed. On the other hand, the camera sensor generally has the advantage of farther visual range and better prospective data than photoelectric sensors, which is more effective to anticipate raceway conditions and then make quick turning or cut corners. Furthermore, the CMOS camera sweeps array sampling points as a whole image, which means it can offer more detailed data. However, large amounts of data require much memory storage space and longer processing time. Another problem usually caused by camera sensors is the background light interference, which makes some negative effects for detection accuracy.

In this system a kind of CMOS camera is used with photosensitive sensor in it, which has the characteristics of high resolution(314×291 valid pixels), low power, 12V DC supply and 50Hz scanning frequency (20ms per frame). Under normal circumstances, it samples 25 images per second and each

image is divided into two frames, so called the odd/even frame. In test, the signals output from the camera sensor are shown as follows:

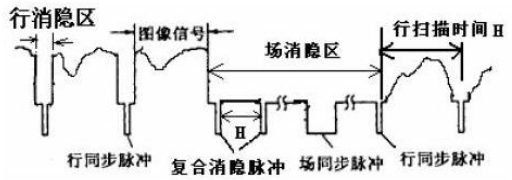


Fig. 1. Original video signals with field and line blank areas

Apart from all the invalid blank signals in the video signal, valid signals are selected by IC LM1881 and then go into A to D convert part of MCU. After the conversion, valid analog signals are turned into digital values shown as follows:

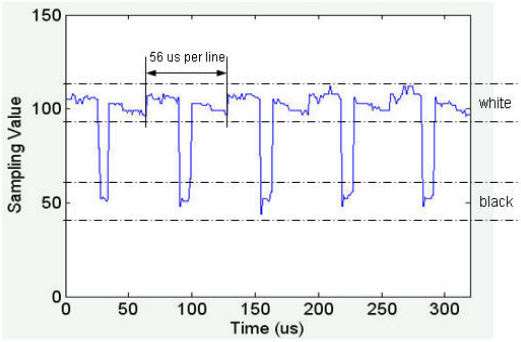


Fig. 2. Sampling value of image frames

In this picture, we can see two threshold bands: higher value (from 95 to 115) represents white and lower value (from 40 to 60) represents black. Obviously, the white part of the image is the raceway background and the black is the middle guiding line, while all the other values could be ignored. The width and reference value of the threshold band are due to the ambient light conditions and different camera models.

The processes of a system with MCU embedded can be defined as four basic facets: sensing, analysis, command and execution. In order to make a sensitive dynamic response and get a high controllability, it is significant to improve system processing speed, especially the sensing process which is the slowest facet of all. There are two aspects affecting the sensing speed most: camera sampling process and A/D conversion process. Firstly, the camera receives images at a speed of 20ms per frame associated with its fixed frequency, so the whole process continues a period of 20ms as well as the maximum mechanical action frequency—50Hz. Secondly, the speed of A to D conversion part inside the MCU is relatively slower than MCU core. Assuming that the MCU worked at a normal speed of 24MHz, and the A/D conversion was slower, the conversion period must last at least 14 A/D clock cycles including 2 initial sample cycles, 2 programmed sample cycles, and 10 resolution cycles. So it could probably achieve a resolution of 48 points per line in 56us. This resolution is very enough for recognizing the position of middle line for tracking, but not enough for recognizing the start point for recording laps as its white gap is too narrow to detect. Detecting the start point shown as Fig. 3 is highly required according to the new competition rule. The problem is that images with too low resolution cannot be recognized the two white gaps between start line and middle line. The gap is the most significant specific character of the start point different from the black crossing which may be confused to recognize. To this problem, one solution is to improve the horizontal resolution of the image. Nevertheless, it is useless to get too high resolution as it cost much storage space and processing time.

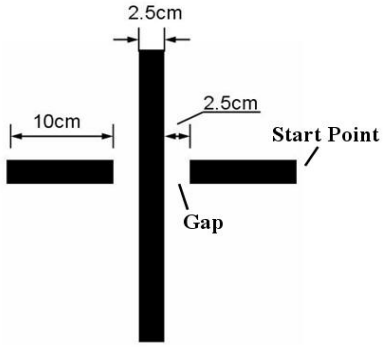


Fig. 3. The start point and middle line

For the sake of improving horizontal resolution, one method is to overclock the MCU bus clock by changing the value of bus clock register. According to the MCU speed formula: BusClock=OSCCLK*(SYNR+1)/(REFDV+1)，it can be overclocked to 48MHz by setting the SYNR and REFDV control bit. However, overclocking would probably make MCU work unstably, so it is not the best choice. Referring the datasheet instruction in Reference [3], it can be found that during A to D conversion, the 10 A/D clock cycles of the resolution period may be unnecessary. That time can be cut out by software method so that converting period is shortened. The general idea of confirming one A/D conversion to be accomplished was to wait the completion signal by checking the ATDxSTAT0 bit. But the waiting time including resolution period is too redundant so it is changed as follows:

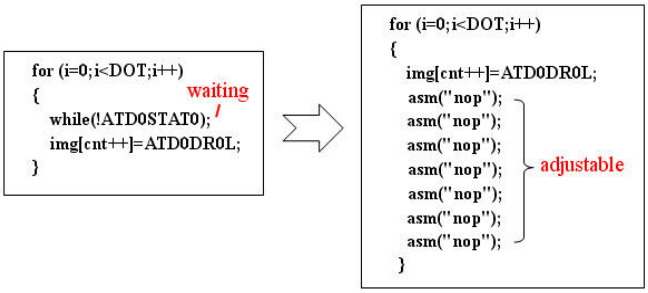


Fig. 4. The “nop” statements instead of waiting statement

The “nop” statement executed in one cycle is inserted into loop program for several times to replace the waiting statement. By adjusting number of the “nop” statements, we can control the delay precisely while the conversion result is correct as well, so that the horizontal resolution can achieve 93 points per line at most without overclocking by this method. Taking into account the high resolution and small data size, we achieve a

resolution of 80 points per line and 45 lines per frame, which are 3600 pixels.

III. IMAGE PROCESSING TECHNOLOGY

Image processing is to selectively simplify a large amount of data sampled by sensors and then analyze the valid information. Because of the high image resolution and continuous raceway, the original redundant data cost lots of storage space and processor resources, hence it's necessary to save storage space by compressing data, keeping valid information and then get rid of noise interference. In the process the middle line should be acquired to guide the car tracking along the raceway, so the image processing subroutine is to pick up the black middle line, throw discrete black points and then calculate car's position relative to the raceway.

The image data are stored in a M×N array. In order to simplify those data, all grey value data should be changed into binary (0,1) distinguished by voltage threshold shown as Fig. 3. The black middle line and white background have different reflectance, that is to say different colors have different voltage levers by camera detection. After binary approach the storage space will be saved a lot. Considering there may be some discrete black points similar with middle line in color, interference might occur during middle line acquisition. Therefore, a necessary step before acquisition is filtering discrete points.

A. Filtering Method

a. Median Filtering Method (MFM)

In this method we calculate the value of each black point and its surrounding points for average, then checking the valid black points whether belong to the middle line by means of comparing average value with threshold. The accuracy of the filtering depends on how many surrounding points we choose. If chosen too few points, some large discrete black points wouldn't be identified or filtered, while on the other hand, too many points would lead to heavy processor load and what's more, affecting start point detection. Experiment demonstrated that the best surrounding zone was shown as follows:

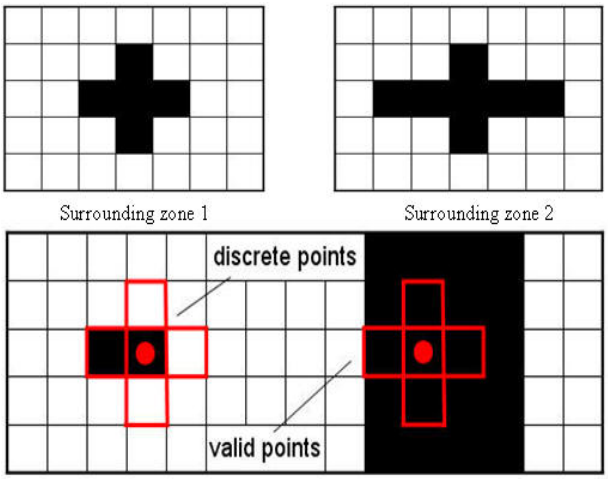


Fig. 5. The optimized surrounding zones of MFM

b. Continuity Judgment Method (CJM)

As it is known that the raceway is continuous so that the column position of every two black points will not be too far away. If it is, the position value of thoss points is obviously incorrect and should be replaced by former one's position value or just thrown.

B. Guide Line Acquisition

The next step is to acquire middle line position, which requires characteristics of fast calculation speed, stable accuracy and good anti-interference capability.

a. Progressive Scanning Method (PSM)

This method is to scan the image date from top to bottom or upside down to find the point which meets the black threshold band. Then record the column number of the first black point j_0 and total number of the black point k. Finally we can get the center position of this line by this formula:

$$j = j_0 + k/2 \quad (0 \le j \le 80) \tag{1}$$

After scanning from top to bottom all the center position numbers are stored in a 1-dimension array. This is a very simple array containing all valid information, and the size of the image data reduces extremely from 3.6K to 45 bytes. In addition, the first black point should be recognized precisely. It is also very convenient to recognize the start point because it has already got the number of black points every line (k) during scanning which is one of the judgment factors of the start point.

b. Regional Search Method (RSM)

This method scans the image from bottom to top. MFM is used as well as PSM to decide the first center position of the bottom line. Then from the last line but one we scan totally 7 points on both sides of the point based on the center position of the former line, comparing those points and find minimum one as the new center position of this line. So dose all the upper lines until to the top. This method works according to the continuous principle of middle line. Its advantage is faster scanning and good anti-interference capability, while the disadvantage is that without whole line scanning (only 7 points per line) it can't detect the start point. However, this method is much better than the above one if there was no requirement in detecting start point.

c. Edge Capturing Method (ECM)

This method works according to the principle of grey value derivative. The image is scanned from left side to right every line and check the sudden falling or raising trend of the data. If there is such a specific trend like falling-keeping-raising which is the reflection of the black middle line under the white background, the column numbers are recorded where it falls and raises, then get the center position by calculating them for the average. The great advantage is that by this method it is unnecessary to set a fixed threshold band for data value but only to consider the derivative, so that the light interference could be ignored and it is no longer to adjust the threshold to meet different raceway environment.

The tracking curve extracted from image is shown as follows restored by MATLAB:

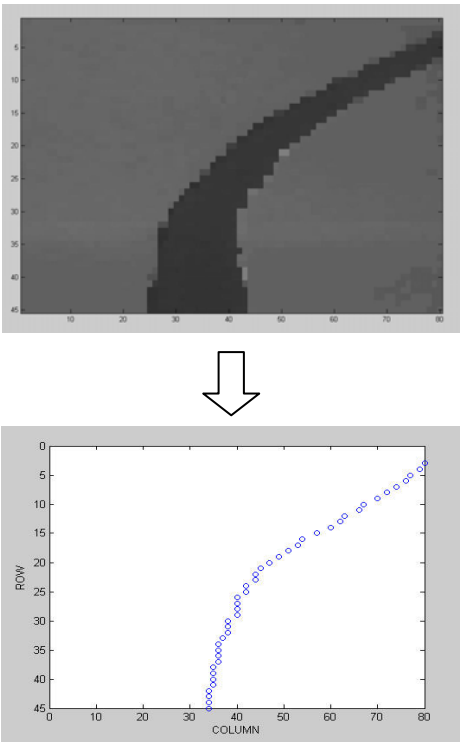


Fig. 6. Image restoration and track extraction

IV. CONTROL STRATEGY AND OPTIMIZED ALGORITHM

A. Image Shape Adjustment

Because of visual prospective, the camera shoots a trapezoid image with a width proportion of 2.5:1 from top to bottom. In order to reflect the real curvature of the middle line, it is necessary to shape trapezoidal image into rectangular one. However, it is such a complex and redundant process of transforming all the image data. A simple way is to transform the array elements of center position every line. Define the array as Path[i] and do shape adjustment as follows:

$$path'[i] = (path[i] - \frac{n}{2}) \times (2.5 - 1.5 \times \frac{i}{m}) + \frac{n}{2} \qquad (2)$$

Where i is the line number and m,n are the length and width of the M×N array.

Another solution is to adjust shape later in turning control algorithm, which will be more effective but a little more difficult.

B. Turning Measurement

It is demonstrated that the turning of the steering wheel are governed by two inputs: CURVE and OFFSET. The former one reflects the raceway curvature and the turning trend. The later reflects the body shift of the car relative to the middle line, and plays a role of fine-tuning to rectify direction in straight way. The CURVE is calculated as follows:

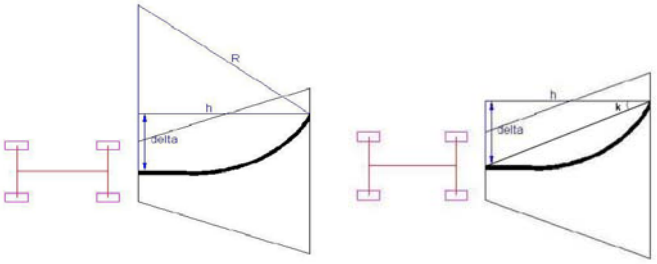


Fig. 7. The curvature and slope calculation

$$curve = \frac{1}{R} = \frac{2delta}{delta^2 + h^2} \qquad (3)$$

$$k = delta / h \qquad (4)$$

Where *curve* is the curvature of the turning and k is the slope of it, *delta* is middle line deviation from top to bottom and h is valid line distance.

The system uses integer to express all variables. It can be seen in Eq.(3) that the CURVE value is usually small and will have great error if expressed by integer. Considering the slope calculation of the curve as Eq.(4), which is defined as the ratio of *delta* and h, is much simpler than curvature calculation. Furthermore the slope k can be substituted by *delta* if the number of valid lines has been determined. Experiment proved that the slope k expressed curvature more accurately than CURVE by the integer DELTA.

The variable OFFSET can be measured by the deviation between center point of the bottom line and physical center of the smart car, shown as follows:

$$offset = path'[m] - n/2 \qquad (5)$$

Where m is the row number of bottom line and n is the column number of the image width.

To sum up above, it is summarized that the turning angle can be defined as follows:

$$angle = f(delta / h, offset) \qquad (6)$$

Where f is a function including two variables.

C. Turning Control Strategy

According to the Eq.(6), the turning angle is a function of variables DELTA and OFFSET. For the requirement of system sensitivity and robustness, this function can be various if chosen different algorithms.

a. The Non-linear P Algorithm with Floating Dead Zone (NPA)

In PID control algorithm, the P, I, D respectively represent the parameter of proportional, integral and derivative. For the smart car system, the I control part can be omitted because it is unnecessary to consider former conditions of the track. The D control prejudges raceway turning trend and rectify turning in advance. But in test it is found that D control doesn't work obviously the same as P control so that it is omitted either. As a

result, the single P parameter is enough for turning control using the following formula:

$$angle = K_1 \times delta / h + K_2 \times offset \quad (7)$$

Where K_1 and K_2 are magnification parameters of slope and offset. The larger K_1 is, the more obviously cutting curve the car appears; while the larger K_2 is, the less obviously cutting curve appears but better tracks the raceway. Generally, slope plays the main role of contributing turning so that K_1 is larger than K_2.

Usually it is difficult to decide proper parameters in theory so that it needs some practice. Experiments show that the actual angle of steering motor is non-linear with the slope k in different angle zones. For the sake of smooth running in straight and fast turning in curve, it is necessary to use different parameters K_1 for zones.

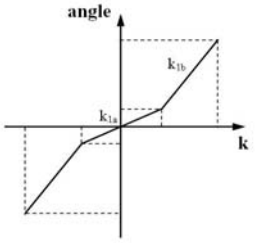
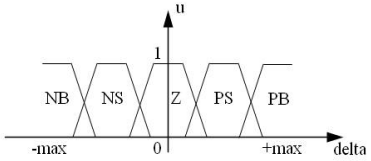


Fig. 8. The non-linear parameter zones of slope k

The steering motor executes actions at the frequency of 25Hz or 50Hz the same as camera frequency. Such a high frequency of turning actions may cause severe vibrations. To avoid those vibrations, a floating dead zone of turning angle is set up, which defines the latest turning angle as the benchmark. If a new angle value was no bigger or smaller than zone limits, the tiny action would be ignored, otherwise the action would be executed and the benchmark would be replaced by the new value. It is called floating dead zone because the zone is frequently changed when a new benchmark appears. Pay attention to the zone range, it should be chosen properly because too narrow range will lead to dissatisfactory anti-vibration result while too wide range may cause oscillating around straight path rather awkwardly.

b. Fuzzy Control Algorithm (FCA)

This algorithm is usually adopted in smart cars using photoelectric sensors. There is only one row of sensors in front of car, so it cannot judge turning trend of the raceway in advance. The inputs to the algorithm are body offset and its derivative, and the output is turning angle. But for forward-looking camera sensors, this algorithm can also be used with the input of DELTA and OFFSET. Here shows the look-up rule table and membership functions representing the fuzzy controller:



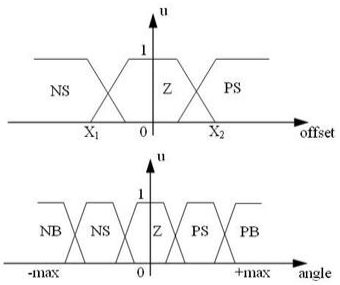


Fig. 9. The membership functions of the fuzzy controller

TABLE I
LOOK-UP RULE OF FUZZY CONTROLLER

offset \ delta	NB	NS	Z	PS	PB
NS	PB	PS	Z	Z	NB
Z	PB	PS	Z	NS	NB
PS	PB	Z	Z	NS	NB

Product Inference for fuzzy reasoning rules:

$$\mu_k(angle) = \mu_i(delta) \times \mu_j(offset) \quad (8)$$

Where i = NB,NS,Z,PS,PB; j = NB,Z,PB; k = NB,NS,Z, PS,PB

Finally the turning angle value is calculated with the weighted average method:

$$angle = \frac{\sum U \times \mu_k(angle)}{\sum \mu_k(angle)} \quad (9)$$

Comparing the two control algorithms above, the NPA is easier but need more test to determine parameters K_1 and K_2, while the FCA is a little complicated but need less experiments and more universal.

D. Speed Control Strategy

Considering that the raceway consists of all kinds of straights and turnings, the car should adjust its speed rapidly associated with road conditions. In order to accelerate or brake as fast as possible, it needs speed measurement feedback. By comparing the real-time speed and expectative speed shown as the flowchart in Fig.12, it can be decided when to accelerate or to break.

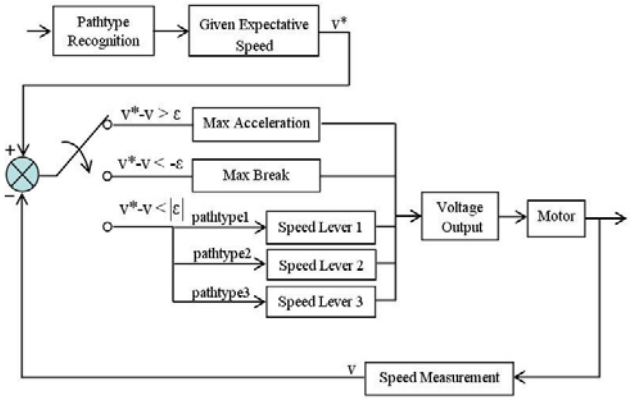


Fig. 10. Flowchart of speed control

E. *Pathway Memory Method*

According to the new rule, the smart car is allowed to run two laps in formal competition and the better lap time is the final result. Because of the rule, the raceway memory method can be operated like this: recording raceway information on the first lap and improving running strategy on the second. There are the Whole Memory Method (WMM) and Partial Memory Method (PMM) that can be chosen differently based on the memory restriction of the MCU system. The WMM is to record all information of the lap including straights, turnings, crossing and start point. But it cost too much data storage space so that PMM is more available. Using the Straight Memory Method (SMM, a kind of Partial Memory Method), all straight information could be stored so that the car can accelerate in maximum and slow down near turning in advance.

The turning of the car is due to many factors such as the camera perspective sight, steering motor performance, suspension system, weight and gravity center, tyre gripping force, raceway friction coefficient, real-time speed, braking force and so on. Generally, the smart car shouldn't exceed a certain maximum speed according to the turning curvature for fear of running outside. For instance, the car often picked up too much speed in straights and then couldn't slow down fast enough for the sharper curves, while using the SMM it can even exceed the maximum speed in straights because it can recognizes the turning and break in advance.

In the SMM there are some variables to be memorized. On first lap we use the timing function of the MCU to record straight length. Then the time and average speed of every straight way are stored in an array one by one, ended by the signal of start point. On the second lap the MCU system reads memorial information to recognize the start and end position of the straights. Finally the car accelerates most at long straights and breaks down hard in advance to turn.

There are some difficulties to be solved while using SMM. The key is to find start and end position of straights precisely: it can be identified as straight when the turning angle is smaller than a threshold, and the MCU timer starts timing until a big angle appears which means straight over. Unfortunately, this method is too ideal while in practical conditions the car may oscillate around straight path so that there maybe mistaken position judgments of start and end as well as mistaken orders. So as a precondition of using SMM, it is necessary to set proper parameters to make the turning smooth and sensitive.

To improve racing speed, it is useful by using SMM if there are many straights in the lap. Though this method works, it has its obvious limitations. For laps with many turnings or other kinds, it may be not useful enough. The Turning Memory Method (TMM) or Whole Memory Method (WMM) may be better.

V. CONCLUSIONS

In this paper, a self-tracking smart car with MCU system embedded is illustrated, including image acquisition, noise filtering, shape reforming, PID control, fuzzy logical control and memory methods of pathway. A matrix CMOS camera is used as eyes of the car. Considering the problem of camera sampling frequency limit—50Hz, the algorithm achieves up to 80 sampling points per line per frame without overclocking (at a normal frequency of 24MHz). Therefore, the image resolution is highly improved, which makes it easier to recognize trace and more accurate in checking start point. The smart car also has autonomous functions of tracking, speed adjusting and strategy analyzing, which make it much faster and more stable to complete the diverse competition.

The Smart Car Competition is a comprehensive and creative competition involving subjects of automatic control, image recognition, sensing technology, electronic and electrical engineering, computer science, mechanical engineering, vehicle engineering and so on. These approaches are generic and can be readily extended for the implementation of similar systems such as intelligent robot arms, automation platforms and urban rail transportations. In order to improve its intelligence ability, some more works should be done:

1. Intelligence of camera visual range: adjustable perspective visual range according to different raceway conditions (e.g. smooth turning, sharp turning and "S" turning). So the car can do better in cutting corners.
2. Intelligence of raceway memory method: stronger memory capability to satisfy storing all the information of the lap. So the car can guide it-self automatically.
3. Intelligence of strategy self-training: adjust system dynamic parameters automatically according to the passed road and running conditions. So the car can perform better by optimized strategy.

ACKNOWLEDGEMENT

The author thanks Yunhai Yang, Zhejiang University, for offering important advices and digitizing the images of the smart car as well as Bin Yu, Longxiang Zhu for hardware and software assistance.

REFERENCE

[1] Motorola Inc, "MC9S12DT128B Device User Guide", Original Released on 18 June 2001.

[2] Motorola Inc, "ATD_10B8C Block User Guide", Original Released on 21 Feb 2003.

[3] Motorola Inc, "ECT_16B8C Block User Guide", Original Released on 18 Jul 2002.

[4] Yunhai Yang, Qi Zhou, Liang Lv, Team Jiebao's Technology Report of the 2nd National Undergraduate Intelligent Car Contest, 2007, unpublished..

[5] Yingjun Lv, Anling Xu, Cancan Chong and Maoyong Cao, "Application Analysis of the Linear CCD in the Path Recognition System", Proceedings of the 2007 IEEE, International Conference on Integration Technology:pp151-154.

[6] Mark J. Embrechts, Frank Dicesare, Mark J. Luetzelschwab, "Fuzzy Logic and Neural Net Control for the 'Smarter Car' ", IEEE International Conference on Volume 1, 22-25 Oct. 1995 Page(s):371 - 376.