MVA '96 IAPR Workshop on Machine Vision Applications, November. 12-14, 1996, Tokyo, Japan

REAL-TIME 3D FEATURE EXTRACTION HARDWARE ALGORITHM WITH FEATURE POINT MATCHING CAPABILITY

Eiichi Hosoya * Takeshi Ogura * Mamoru Nakanishi †

* NTT System Electronics Laboratories † NTT Electronics Technology Corp.

Abstract

This paper proposes a real-time 3D feature extraction hardware algorithm with feature point matching capability between neighboring frames, which realizes 3D tracking of moving objects. This hardware algorithm is based on a 3D voting method. Both the 3D voting and the feature point matching are directly carried out through highly parallel processing by content addressable memory (CAM) in real time. For the 3D voting, the CAM acts as a PE (Processing Element) array that performs highly parallel processing. For the feature point matching, the CAM executes a highly parallel processing of nearest neighbor search. Simulations of CAM hardware size, processing time and accuracy show that real-time 3D feature extraction and feature point matching can be achieved using a single CAM chip with current VLSI technology. This CAM-based algorithm promises to be an important step towards the realization of a real-time and compact 3D tracking system for moving objects.

1 Introduction

The 3D tracking of moving objects requires both 3D feature extraction and feature point matching between neighboring frames. For 3D feature extraction from multiple view points, several algorithms using 3D voting have been proposed [1]–[4]. These 3D voting based algorithms have several advantages, the most important one being their robustness to noise. However, these algorithms can not achieve feature point matching, which is essential for moving object tracking. Feature point matching between neighboring frames can basically be realized by nearest neighbor search. However, nearest neighbor search is time consuming and costly.

*Address: 3-1, Morinosato Wakamiya, Atsugi-Shi, Kanagawa Pref., 243-01 Japan. E-mail: hosoya@aecl.ntt.jp, ogura@aecl.ntt.jp

†Address: 394-1, Onna, Atsugi-Shi, Kanagawa Pref., 243 Japan. E-mail: mamoru@center.nel.co.jp

In order to attain a real-time and compact 3D tracking system for moving objects, we propose a 3D voting based feature extraction hardware algorithm, that can also realize feature point matching between neighboring frames in real time. Both the 3D voting and the feature point matching are directly executed by content addressable memory (CAM) in parallel. For the 3D voting, the CAM acts as a PE (Processing Element) array that performs highly parallel processing. For the feature point matching, the CAM executes a highly parallel processing of nearest neighbor search.

The performance of the CAM-based algorithm, i.e., the CAM hardware size, the processing time and the accuracy of 3D tracking for moving objects, was evaluated by simulation. The results show that real-time 3D feature extraction and feature point matching can be achieved using a single CAM chip with current VLSI technology.

Section **2** describes the 3D feature extraction hardware algorithm with feature point matching capability. Section **3** discusses CAM hardware size and processing time and shows the simulation results for 3D tracking for moving objects.

2 3D Feature Extraction with Feature Point Matching Capability

2.1 CAM-based 3D Feature Extraction

The CAM-based 3D feature extraction algorithm is schematically shown in Fig. 1. This algorithm is based on a 3D voting method. In the figure, R is the number of cameras, V is the maximum number of feature points in an image, and N is the side size of 3D voxel space. The 3D voxel space is sliced as shown in the figure. On every sliced plane, two kinds of processing are executed. One is the voting, which is done to count the number of back-projection lines which have the same coordinates in the sliced plane. After the voting, a maximum value updating process for every basic back-projection line is executed. These two processes are executed by the CAM and

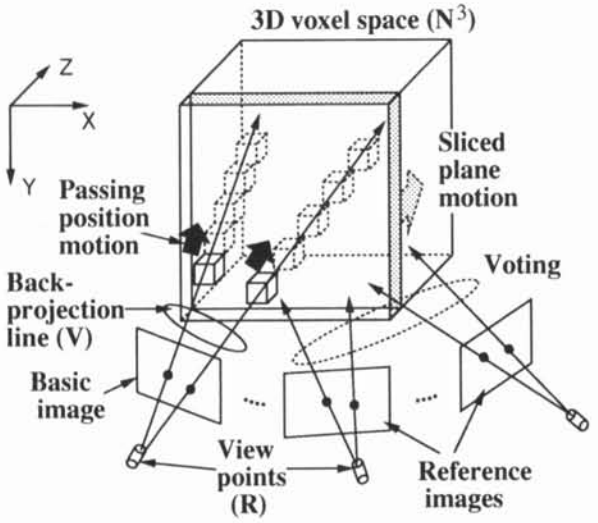

Fig. 1. 3D voting algorithm.

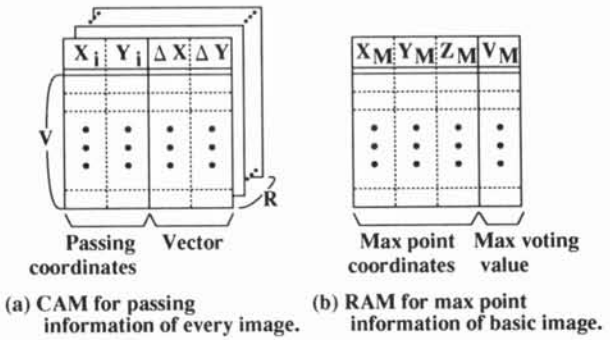

Fig. 2. Memory configuration.

are iterated for all sliced planes. Each feature point on the basic image exists at the coordinates where the voting result is maximum.

In this hardware algorithm, the CAM acts as a PE array that calculates the coordinates of every back-projection line in each sliced plane. Figure 2 shows the memory (CAM and RAM) configuration. Every image has a CAM of V words, and the basic image has a RAM of V words. In the CAM, passing information on the back-projection lines is stored. A CAM word corresponds to one back-projection line, and keeps passing coordinates (X_k, Y_k) on the current sliced plane (Z=k) and line vectors for X, Y coordinates (ΔX, ΔY). Using these values, the passing coordinates (X_{k+1}, Y_{k+1}) on the next sliced plane are calculated as

$$X_{k+1}=X_k+\Delta X,\ Y_{k+1}=Y_k+\Delta Y.$$

These calculations are executed in parallel for all back-projection lines by the CAM. The performance of this hardware algorithm has already been evaluated and reported [4]. 3D coordinates of many feature points (up to about 1,000 points) can be obtained in real time.

2.2 Feature Point Matching between Neighboring Frames

Highly parallel feature point matching between neighboring frames is also achieved by using CAM. The matching process is executed at the initializing of CAM for the basic image, as shown in Fig. 3. After the 3D feature extraction process for the previous frame, all back-projection line coordinates on the last sliced plane are stored in the CAM. Using these last coordinates and the initial coordinates of the next frame on the same plane, the nearest neighbor can be found through a highly parallel search carried out by CAM function. The obtained nearest neighbor is considered as the same feature point. A schematic diagram of the matching and the extraction is shown in Fig. 4. In order to execute the matching efficiently at the last sliced plane, processing direction for the extraction is reversed for each frame.

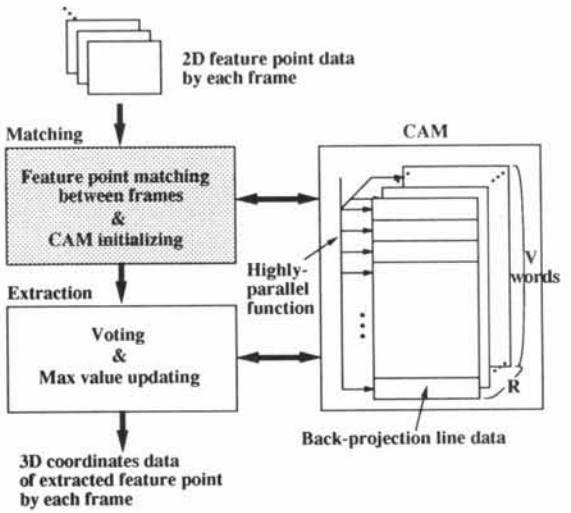

Fig. 3. CAM-based processing flow.

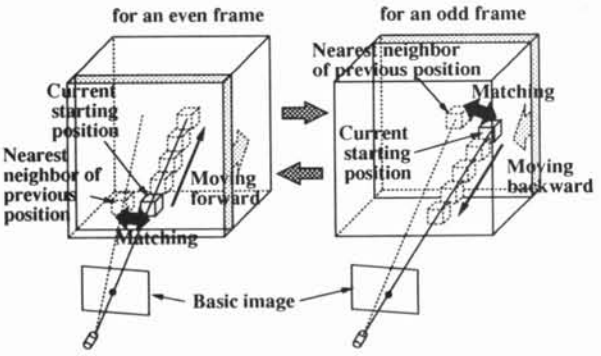

Fig. 4. Feature point matching.

For matching by nearest neighbor search, the distance between the target feature point of a new frame and every feature point of the previous frame must be calculated. Therefore, the distance field is

added to the CAM, which performs a highly parallel calculation of the distance. In this study, we used the Manhattan distance D_i, as expressed by the equation

$$D_i = |X_{\mathrm{NEW}} - X_i| + |Y_{\mathrm{NEW}} - Y_i|, \qquad (1)$$

where $(X_{\mathrm{NEW}}, Y_{\mathrm{NEW}})$ are the feature point coordinates of the new frame and (X_i, Y_i) are the old coordinates stored in the CAM.

The calculation of D_i and the matching process are necessary only for the basic image. For other reference images, all initial data of back-projection lines are written into the CAM sequentially. Figure 5 shows the details of the processing flow for matching in the basic image.

Step1: For each initial coordinates of new frame $(X_{\mathrm{NEW}}, Y_{\mathrm{NEW}})$, calculate Manhattan distance D_i to every coordinate of the previous frame on the CAM (X_i, Y_i). All D_i's can be obtained simultaneously by using highly-parallel CAM function.

Step2: Execute a parallel search for the word with the minimum D_i value.

Step3: Store the initial coordinates of the new frame $(X_{\mathrm{NEW}}, Y_{\mathrm{NEW}})$ and $(\Delta X, \Delta Y)$ at found word. The nearest neighbor points of previous and new frames have the same index.

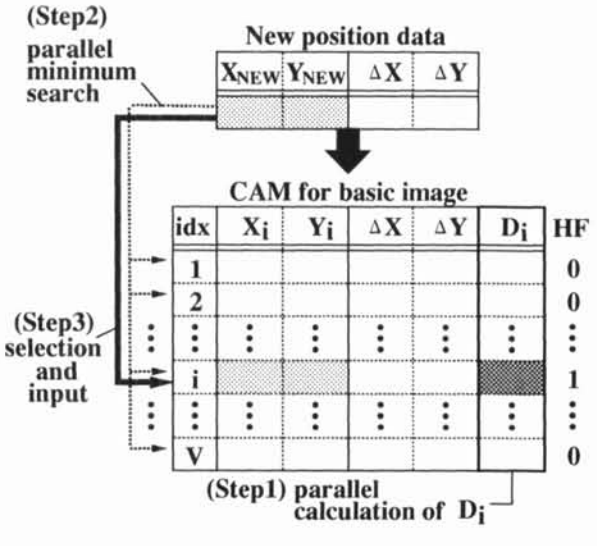

Fig. 5. Processing flow for matching.

The above hardware algorithm can be executed in parallel by CAM function. In **Step1**, it is necessary to calculate eq. (1) for all words in the CAM. In **Step2**, minimum search is also carried out for all words. Both **Step1** and **Step2** are executed without dependence on the number of CAM words.

3 Performance Evaluation

3.1 CAM Hardware Size

The main hardware for this hardware algorithm is the CAM. The number of CAM words depends on V and R. The word length depends on N. The CAM hardware size for this hardware algorithm is 68[bit] $\times$ $(V \times R)$[words], where N is 256.

On the other hand, a CAM chip with the capacity of 84[bit] $\times$ 4[Kword] has already been developed [5]. This is enough capacity for this hardware algorithm. All back-projection line data for all images can be stored in a the single CAM chip. This means that the hardware algorithm can be realized on a single board.

3.2 Processing Time

The estimated processing times for feature extraction and matching are shown in Table I. For the estimation, it was assumed that all data are stored in a single CAM chip, the number of cameras R=6, 3D voxel space size N=256, and the clock speed is 25 MHz.

Compared with sequential processing, the matching time decreases from $O(V^2+VR)$ to $O(VR)$, and the extraction time is reduced from $O(RV^2N)$ to $O(RVN)$; that is, an improvement in processing speed of at least V times is achieved. As shown in the table, it is possible to execute extraction and matching in real time up to about 200 points.

Table I Estimated processing times using a single CAM chip.

Number of points V	50	100	200	
Extraction time	8.8	15.0	27.3	[ms / frame]
Matching time	0.8	1.6	3.2	[ms / frame]
TOTAL	9.6	16.6	30.5	[ms / frame]

(R=6, N=256)

3.3 Feature Extraction and Tracking Performance

Feature extraction and tracking performance for moving objects was preliminarily evaluated by simulation. In the simulation, target objects were many moving points in the 3D space. Each point had initial coordinates (x_0, y_0, z_0) and vector $(\Delta x, \Delta y, \Delta z)$ for moving, which were created at random. The 3D space was limited and these points were reflected at the boundary. For every frame, 3D feature extraction and feature point matching was executed. After 10-frame processing, we obtained the probability of

success in extracting feature points and matching between continuous frames.

The dependence of the probability P on the number of feature points is shown in Fig. 6. In the figure, S is the $\max(|\Delta x|+|\Delta y|+|\Delta z|)$ of moving feature points. In this case, R=6, N=256, V=10 to 200. The probability P is more than 85% when the number of feature points is 100 and S is 5. This is good enough for practical applications.

Figure 7 shows an example of the extraction and tracking of moving points for 20 frames, where V=10 and S=7. In this case, the probability P is about 98%. Within the 20 frames, there appear two feature points that correspond to wrong indexes, in two frames, as shown in Fig. 7 (3),(4). These errors were caused by feature point crossing.

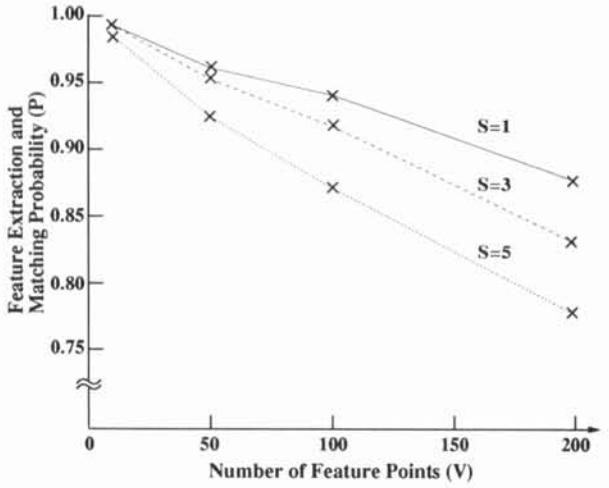

Fig. 6. Extraction and matching performance; The probability is the average of 5 times.

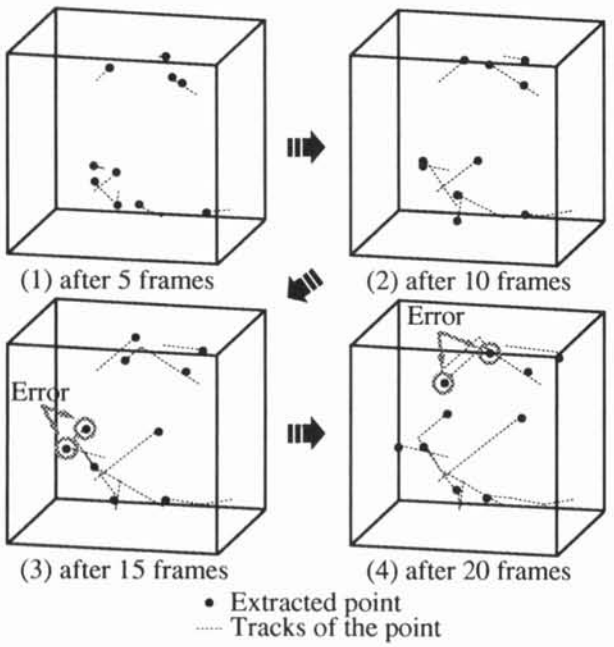

Fig. 7. Example of extraction and tracking.

4 Conclusion

We proposed a real-time 3D feature extraction hardware algorithm with feature point matching capability between neighboring frames, which realizes 3D tracking of moving objects. In this algorithm, highly parallel processing can be carried out in both the 3D voting and the feature point matching by effectively using the highly parallel processing function of CAM. Simulation results of matching performance indicate that, with the proposed hardware algorithm, the probability of success in extracting and matching is enough for some practical applications such as human motion capturing. Based on the estimation of hardware size and the required processing time, it was shown that real-time 3D feature extraction and feature point matching can be achieved using a single CAM chip with current VLSI technology.

In future work, we will evaluate the performance of this algorithm using real images, and develop a single board system for the 3D tracking of moving objects.

References

[1] T. Hamano, T. Yasuno and K. Ishii, "Structure from Motion by Voting Algorithm," *IEICE Trans. on Information and Systems*, Vol. J75-D-II, No. 2, pp. 342-350 (1992).

[2] S. Kawato, "3D Shape Recovery by Octree Voting Technique," *Videometrics(SPIE)*, Vol. 1820, pp. 131-140 (1992).

[3] K. Tsuchiya, S. Hashimoto and T. Matsushima, "A Pixel Voting Method to Recover 3D Object Shape from 2D Images," *Proceedings of MVA '94*, pp. 111-114 (1994).

[4] E. Hosoya, T. Ogura and M. Nakanishi, "A High Resolution Highly-Parallel Hardware Algorithm for Realtime Extraction of 3D Information," *Proceedings of the 1995 IPSJ Fall conference*, No. 51, 2S-6 (1995).

[5] T. Ogura, et al "A 336-kbit Content Addressable Memory LSI for Highly Parallel Image Processing," *Custom Integrated Circuits Conference '96*, 13.4.1, pp. 273-276 (1996).