Pixel Selection and Intensity Directed Symmetry for High Frame Rate and Ultra-Low Delay Matching System

SUMMARY High frame rate and ultra-low delay matching system plays an increasingly important role in human-machine interactive applications which call for higher frame rate and lower delay for a better experience. The large amount of processing data and the complex computation in a local feature based matching system, make it difficult to achieve a high process speed and ultra-low delay matching with limited resource. Aiming at a matching system with the process speed of more than 1000 fps and with the delay of less than 1 ms/frame, this paper puts forward a local feature based matching system with field-programmable gate array (FPGA). Pixel selection based 4-1-4 parallel matching and intensity directed symmetry are proposed for the implementation of this system. To design a basic framework with the high process speed and ultra-low delay using limited resource, pixel selection based 4-1-4 parallel matching is proposed, which makes it possible to use only one-thread resource consumption to achieve a four-thread processing. Assumes that the orientation of the keypoint will bisect the patch best and will point to the region with high intensity, intensity directed symmetry is proposed to calculate the keypoint orientation in a hardware friendly way, which is an important part for a rotation-robust matching system. Software experiment result shows that the proposed keypoint orientation calculation method achieves almost the same performance with the state-of-artist intensity centroid orientation calculation method in a matching system. Hardware experiment result shows that the designed image process core supports to process VGA (640×480) videos at a process speed of 1306 fps and with a delay of 0.8083 ms/frame.

key words: high frame rate and ultra-low delay image processing, local binary feature based matching, keypoint orientation calculation, FPGA

1. Introduction

Real-time vision systems play an important role in everyday life. The general real-time vision system processes 60 fps videos within 16 ms, which is enough for the most common cases when the target object moves slow and the machine does not need immediate reactions in milliseconds’ level. However, with the high pace of technology development, more and more human-machine interactive applications call for higher frame rate and lower process delay to process fast moving object within milliseconds, which leads to an inevitable need for the high frame rate and ultra-low delay vision system. Among various of image processing techniques, local feature based matching plays an important role in many computer vision technologies, such as object tracking [1], 3D reconstruction [2], and camera localization [3]. These computer vision technologies will contribute to the further human-machine interactive applications, such as projection mapping [4], automatic driving [5], and virtual reality. Therefore, high frame rate and ultra-low delay matching system draws the great attention of people.

Generally, a local feature based matching system consists of three parts: 1) keypoint detection; 2) descriptor generation; and 3) descriptor matching.

1) Keypoint detection: FAST [6] is commonly used for keypoint detection in real-time systems because of its fast computational ability. However, FAST does not provide a good measure of cornerness, which makes it not repeatable enough. Harris corner detector [7], which defines a mathematical form to find the corners according to the nature of the corner, is not fast as FAST, but provides a good measure of cornerness and is more repeatable. The results in Heinly’s work [8] also shows that Harris does quite well when coupled with many descriptors.

2) Descriptor generation: Scale-invariant feature transform (SIFT) [9] and speeded-up robust features (SURF) [10] are commonly used for keypoint description because of their good performance. However, the high computational complexity and the large dimensionality of the generated descriptor make it difficult to implement on embedded systems for real-time. In recent years, binary descriptors such as binary robust independent elementary features (BRIEF) [11] and oriented FAST and rotated BRIEF (ORB) [12], can directly build short descriptors with low computational complexity by comparing the intensities of pixel pairs in the surrounding area of the keypoint. Using comparison as the most basic operation and with the high parallel computational ability, BRIEF and ORB is much more suitable for the implementation on FPGA.

3) Descriptor matching: Brute force matcher, which finds one best match for each query descriptor, is commonly used in the implementation of embedded systems because of its simple structure.

In recent years, several works [13]–[16] that try to accelerate the local feature based matching system by FPGA have been presented. So far, no work targets on the high frame rate and ultra-low delay matching system is found. Suzuki’s work [13] and Rao’s work [14] support to process 60 fps videos within 16 ms/frame. The process time of Heo’s work [16] reaches 18 ms/frame, which is far away from the needs of a high frame rate and ultra-low delay system. As a result, the current existing works does not support to process 1000 fps videos within 1 ms/frame.

In order to design a rotation-robust matching system...
with high frame rate (≥ 1000 fps) and ultra-low delay (≤ 1 ms/frame), two problems in the conventional methods need to be solved: 1) sequential structure in the basic matching system; and 2) complex operations in the current keypoint orientation operator.

1) Sequential structure in the basic matching system. In current existing works that accelerate the local feature based matching by FPGA [13]–[16], the systems are generally designed to process only one pixel at one time, and a RAM based sequential matching structure is designed to find the match for each query descriptor by checking the descriptors in the training dataset one by one. With this design, the resource consumption of this system becomes quite small while the processing delay becomes large and the process speed is also at a low level.

2) Complex operations in the current keypoint orientation operator. Keypoint orientation operator is needed in the descriptor generation part when considering the rotation robustness of the matching system. Intensity centroid, firstly presented in Rosin’s work [17], is a fast and accurate orientation operator. ORB [12] uses this method to build a rotation-robust descriptor which is capable of real-time performance. However, several operations which are not friendly for FPGA implementation are included in this method, such as multiplication, division, and inverse trigonometric function. The conventional works [14], [15] try to implement the intensity centroid into FPGA in a hardware-friendly way. Instead of calculating the keypoint orientation through the original equation presented in [12], the conventional works [14], [15] manipulate the equation into another form to calculate the keypoint orientation in an indirect way, which can avoid the division operation. However, the indirect method generates a larger delay, and more multiplication operations are needed. This makes it hard to be capable of the high frame rate and ultra-low delay performance. Symmetry is also an orientation operator presented in Rosin’s work [17], which decides the keypoint orientation based on the corner symmetry. With high parallel computation ability and without using operations such as division and inverse trigonometric function, symmetry is more suitable for FPGA design. However, the orientation calculated by this method is with a range of (0°, 180°), which is not enough for the improvement of the rotation robustness in the matching system.

This paper proposes a high frame rate and ultra-low delay matching system that is robust to rotation, and its real-time hardware implementation on FPGA. Firstly, to design a parallel structure in the basic matching system, pixel selection based 4-1-4 parallel matching is proposed. With a pixel selection based 4-1-4 structure, the whole system is paralleled to process 4 pixels at one time while using only 1-thread resource consumption in the main process. And a register storage based parallel structure is designed in the descriptor matching part, which accelerates the matching process effectively. Secondly, intensity directed symmetry is proposed to calculate the keypoint orientation in a hardware-friendly way from the algorithm level. Assumes that the keypoint orientation will maximize the corner symmetry and it will point to a region with high intensity, intensity directed symmetry uses the operations with less computational complexity and can calculate keypoint orientation with a range of (0°, 360°). In this way, a rotation-robust matching system with high frame rate and ultra-low delay is designed with small resource consumption.

2. Proposed High Frame Rate and Ultra-Low Delay Matching System

2.1 Framework of High Frame Rate and Ultra-Low Delay Matching System

The framework of the whole matching system is shown in Fig. 1. In the keypoint detection part, Harris [7] is used to detect the corner as the keypoint. And the local maximum neighbouring check presented in our previous work [18] is applied to reduce the large delay in Harris [7] caused by the global sorting. In the descriptor generation part, the rotation-robust binary descriptor of each keypoint is generated based on ORB [12]. And there are mainly three important steps in the descriptor generation part for generating the rotation-robust descriptor. First, the orientation of the keypoint is calculated. Next, the patch used to generate the descriptor is rotated according to the orientation. At last, the descriptor generated in the new patch is robust to rotation. In descriptor matching part, the matched keypoint for each keypoint in the current frame of a video stream is found from the template based on the brute force matching. For each detected keypoint in the current frame, a candidate match with the minimum distance in the object template can be found according to the similarity of the keypoint descriptors, which can be measured by the hamming distance. The candidate match is considered as a match only when the corresponding minimum distance is under a threshold.

The positions of the proposed two methods are also shown in Fig. 1. Pixel selection based 4-1-4 parallel matching is proposed to design a parallel structure in the basic matching system. Intensity directed symmetry is proposed to calculate the orientation of the keypoint in a hardware-friendly way from the algorithm level. These two proposals will be introduced in detail in this section.
2.2 Pixel Selection Based 4-1-4 Parallel Matching

Figure 2 shows the conceptual difference of the basic structure in descriptor generation and descriptor matching. In the conventional method, the matching system processes only 1 pixel at one time, and the process of the descriptor matching is designed based on the RAM storage, which makes the whole structure sequential. With this design, the resource consumption of this system is small, but the process speed is slow and the delay becomes large, which cannot be capable in the high frame rate and ultra-low delay system. Pixel selection based 4-1-4 parallel matching is proposed to build a parallel structure of the basic matching system.

There are mainly 2 parts in the pixel selection based 4-1-4 parallel matching. 1) Pixel selection based 4-1-4 thread transformation, which makes it possible to use one-thread resource consumption to achieve a four-thread processing in the descriptor generation and descriptor matching part of the matching system. 2) Register storage based parallel matching, the descriptor matching of the system is designed fully paralleled based on the register storage, which reduces the delay of the descriptor matching effectively.

2.2.1 Pixel Selection Based 4-1-4 Thread Transformation

In the hardware design, it’s difficult to balance the process speed and resource cost with the parallel structure. As shown in Fig. 3, for a system, if it processes 1 data at one time, the resource cost of the system is small, but the process speed is slow. If the system processes four data (more than one) at one time, the process speed becomes 4 times faster while the resource cost becomes 4 times larger. However, in the matching system, it’s found that the descriptor generation and descriptor matching is only needed for the pixel which is a keypoint. Therefore, the idea that making sure there is at most one keypoint in the simultaneously processed four pixels and selecting one appropriate pixel from these four pixels to do the main process, comes into consideration.

Pixel selection based 4-1-4 structure makes it possible to use one-thread resource consumption to do the four-thread processing. In the proposed 4-1-4 structure as shown in Fig. 2, the overall of the matching system is paralleled into four threads, which supports to process four pixels at one time. There are mainly three key points in this method: 1) local maximum neighbouring check; 2) pixel selection; and 3) allocation.

1) Local maximum neighbouring check: In the keypoint detection part, the Local maximum neighbouring check is utilized to reduce the large delay caused by the global sorting in the Harris [7]. Here, it has another function – make sure there is at most 1 keypoint in the simultaneously processed 4 pixels. According to the local maximum neighbouring check, a pixel is a keypoint only when its Harris score is the biggest one in its check block. Figure 4 shows the check blocks of the simultaneously processed 4 pixels in local maximum neighbouring check.

2) Pixel selection: Pixel selection is employed to reduce the redundancy in our designed 4-paralleled structure.
Resource consumption of the descriptor generation and descriptor matching will become 4 times larger when the number of the simultaneously processed pixels increases into 4. However, in the matching system, the process of the other 3 pixels that are not keypoints is redundant when there is at most 1 keypoint in the simultaneously processed 4 pixels. Pixel selection is utilized to reduce this redundancy by selecting one appropriate pixel from the 4 pixels to do the descriptor generation and descriptor matching. Before descriptor generation and descriptor matching, each pixel of the 4 simultaneously processed pixels is distinguished whether it is a keypoint or not according to the flag made for each pixel in the keypoint detection part. The keypoint is selected as the appropriate pixel when there is one keypoint in these 4 pixels. In the conventional method as shown in Fig. 2, in order to keep a pipeline structure, the descriptor is calculated for the coming pixel regardless of it’s a keypoint or not. Here, the same consideration is utilized when there is no keypoint in the 4 simultaneously processed pixels.

3) Allocation: After the descriptor generation and descriptor matching, the corresponding results are allocated to these four simultaneously processed pixels. For the pixel which is a keypoint, the result processed after the descriptor generation and descriptor matching is directly allocated to it. For the pixel which is not a keypoint, the result which represents not matched is allocated to it. Although there is only one pixel processed in the descriptor generation module and descriptor matching module, the throughput of the system is four pixels/clock.

2.2.2 Register Storage Based Parallel Matching

In the descriptor matching part, a parallel matching based on the register storage is put forward to accelerate the process of the descriptor matching. Conventional method stores all the descriptors of the template in a RAM. Data stored in a RAM can only be accessed one by one, which leads to a sequential processing with large delay when finding the match from the template. It’s reasonable to store the data in the RAM when the data length is large, such as Suzuki’s work [13] which implements SIFT (128 bytes for each keypoint) based matching system on FPGA. However, the data length of the binary descriptor is short, and 16-byte binary descriptor is generated in the high frame rate and ultra-low delay matching system, which makes it possible to store all the descriptor data in a register. Register storage has the property that all the data stored in it can be accessed at one time, which makes it possible to design a parallel process.

Figure 5 shows the process flow of the register storage based parallel matching. 64 descriptors selected from the template are stored in a template register. There are mainly three parts in the matching process. Firstly, 64 128-bit xor data between the currently processed descriptor and 64 descriptors in the template register can be obtained simultaneously through the “XOR” operation. Secondly, 64 hamming distances can be obtained simultaneously after applying the “population count” operation on the 64 xor data. In order to achieve a high maximum frequency, 64 128-bit xor data is divided into 4 parts and every 64 32-bit xor data are processed simultaneously. The final population counting result can be obtained after the 4 parts’ population counting results are added up in the “Addition” operation. At last, the minimum distance can be found after obtaining all the distance between the currently processed descriptor and all the descriptors in the template, and a distance threshold is prepared to eliminate the outmatching pairs. Since the 64 hamming distance data can be obtained simultaneously, the process of “find mini-distance” is designed fully paralleled as shown in Fig. 6. The whole matching process is made fully paralleled and pipelined. In this way, descriptor matching module with high process speed and low delay is designed.

2.3 Intensity Directed Symmetry

To make a rotation-robust matching system be capable of high frame rate and ultra-low delay performance, an orientation operator named as intensity directed symmetry, is proposed to calculate the orientation of the keypoint in a hardware-friendly way from the algorithm level. The proposed method mainly has three advantages. Firstly, the orientation calculated through the proposed method is with a range of \((0^\circ, 360^\circ)\). Secondly, it has a high parallel capability. Thirdly, the operations in this method are friendly for the FPGA design. This subsection will introduce the detailed
algorithm and its appropriate hardware implementation.

2.3.1 Algorithm of Intensity Directed Symmetry

Figure 7 gives the conceptual comparison between conventional methods and the intensity directed symmetry. Conventional1 – intensity centroid, defines the vector points from the keypoint to the intensity centroid as the keypoint orientation. Conventional2 – symmetry, assumes that the keypoint orientation maximizes the corner symmetry, and finds the line that bisects the patch around the keypoint best as the keypoint orientation. The proposed method assumes that the keypoint orientation will bisect the patch best and will point to the region with high intensity. Firstly, a line, which bisects the patch around the keypoint best, is found based on the definition of keypoint orientation that the keypoint orientation will bisect the patch around the keypoint best, and this line is called as the pre-orientation \( \hat{\phi} \). Next, a direction is given to the pre-orientation \( \phi \) based on the definition of keypoint orientation that the keypoint orientation will point to the region with high intensity. A directed line consisting of the pre-orientation \( \phi \) and its given direction is defined as the keypoint orientation.

The detailed process of the intensity directed symmetry is as the following:

**Step 1:** Calculate the pre-orientation \( \phi \). The pre-orientation \( \phi \) is defined as the line that bisects the region surrounding the keypoint best. In order to find the pre-orientation, symmetry value \( SV_\phi \) for each hypothesized orientation \( \phi_i \) is calculated. In the conventional method symmetry [17], Rosin rotates the image window by the hypothesized orientation, and the difference of the intensity sum between the two regions that are separated by the x axis is used to measure the corner symmetry. The pixel intensity in the rotated patch is obtained by the bilinear interpolation, which is too complex for high frame rate and ultra-low delay system. Here, instead of rotating the image window and obtaining the pixel intensity by bilinear interpolation, the pixel intensities in two regions (region1 and region2) that separated by the hypothesized orientation \( \phi_i \) in the patch around the keypoint (corner) are directly used to measure the corner symmetry. The symmetry value of a patch for each hypothesized orientation is defined as:

\[
SV_\phi = | \sum I_{region1_{\phi_i}} - \sum I_{region2_{\phi_i}} | \tag{1}
\]

where \( \sum I_{region1_{\phi_i}} \) is the sum of intensities in region1 when hypothesized orientation is \( \phi_i \), \( \sum I_{region2_{\phi_i}} \) is the sum of intensities in region2 when hypothesized orientation is \( \phi_i \). With these symmetry values, the pre-orientation \( \phi \) can be found as which minimizes the symmetry value in all the hypothesized orientations:

\[
\hat{\phi} = \min_{\phi_i} SV_\phi \tag{2}
\]

It’s necessary to mention that 32 hypothesized orientations are prepared to calculate the pre-orientation in the current system. The accuracy of the calculated orientation is proportional to the number of the hypothesized orientations. It’s possible to select a different number of hypothesized orientation according to the needs of the target.

**Step 2:** Give direction to the pre-orientation \( \hat{\phi} \). As shown in Fig. 7, there are 2 directions that are possible to be given to the pre-orientation \( \hat{\phi} \). One of the directions, combined with the pre-orientation \( \hat{\phi} \), forms the directed line \( \hat{\phi}^1 \). The other one of the directions, combined with the pre-orientation \( \hat{\phi} \), forms the directed line \( \hat{\phi}^2 \). The keypoint orientation \( \phi \) is selected from \( \hat{\phi}^1 \) and \( \hat{\phi}^2 \). The line that is perpendicular to the pre-orientation \( \hat{\phi} \) divides the patch around the keypoint into two regions – region3 and region4. Region3 is the region pointed by \( \phi_1 \), Region4 is the region pointed by \( \phi_2 \). The keypoint orientation \( \phi \) is determined by:

\[
\phi = \begin{cases} 
\phi_1, & \sum I_{region3_{\phi_1}} > \sum I_{region4_{\phi_2}} \\
\phi_2, & \sum I_{region3_{\phi_1}} \leq \sum I_{region4_{\phi_2}} 
\end{cases} \tag{3}
\]

where \( \sum I_{region3_{\phi_1}} \) is the sum of intensities in region3 pointed for each hypothesized orientation \( \phi_1 \), \( \sum I_{region4_{\phi_2}} \) is the sum of intensities in region4 pointed by \( \phi_2 \). It’s worth to mention that \( \sum I_{region3_{\phi_1}} \) and \( \sum I_{region4_{\phi_2}} \) have been calculated in step 1, which makes the implementation of this step quite simple.

2.3.2 Hardware Implementation of Intensity Directed Symmetry

In the implementation of intensity directed symmetry, firstly, the sum of intensity in region1 \( \sum f(x,y)_{region1_{\phi_i}} \) and the sum of intensity in region2 \( \sum f(x,y)_{region2_{\phi_i}} \) for each hypothesized orientation \( \phi_i \) are calculated. This process is called as the region sum (RS) calculation. In order to balance the process speed and the resource cost, an asymptotic structure is put forward to calculate the region sum (RS).

Figure 8 shows the conceptual comparison between a general method and the proposed method in the implementation of region sum calculation part:

**Fully parallel region sum calculation:** With a high parallel
computational capability, it’s general to consider a fully parallel structure as shown in Fig. 8 (a). The region sum (RS) for each hypothesized orientation \( \phi_i \) is calculated by:

\[
RS_{region1_{\phi_i}} = \sum I_{region1_{\phi_i}} \tag{4}
\]

\( RS_{region2_{\phi_i}} \) is calculated in the same way with \( RS_{region1_{\phi_i}} \). High process speed can be achieved with this method while the problem of the resource cost comes into consideration. Firstly, the resource cost of \( RS_{PU} \) itself is heavy when the patch used to calculate the orientation is large. The resource cost in region sum calculation will become quite large when all the \( RS_{PU} \) are paralleled. Secondly, the resource cost is proportional to the number of the hypothesized orientation, which leads to large increments in resource consumption when the number of hypothesized orientation increases.

**Asymptotic region sum calculation:** Because of the small change in the number of processing data between two adjacent hypothesis orientations as shown in Fig. 9, an asymptotic structure is put forward to calculate the region sum according to:

\[
RS_{region1_{\phi_i}} = RS_{region1_{\phi_{i-1}}} + DRS_{region1_{\phi_i}} \tag{5}
\]

where \( DRS_{region1_{\phi_i}} \) is the difference region sum in region1 between \( \phi_i \) and \( \phi_{i-1} \). For the first hypothesized orientation \( \phi_1 \), \( RS_{region1_{\phi_1}} \) is calculated according to Eq. (4). \( RS_{region2_{\phi_i}} \) is calculated in the same way with \( RS_{region1_{\phi_i}} \). With the asymptotic region sum calculation, the region sum can be calculated with less resource utilization. Furthermore, there will be a small change of resource consumption when the number of the hypothesized orientation changes.

### 2.4 Hardware Structure of High Frame Rate and Ultra-Low Delay Matching System

Figure 10 shows the demonstration of the high frame rate and ultra-low delay matching system. With PC, high-speed camera, and FPGA, a Camera \( \Rightarrow \) FPGA \( \leftrightarrow \) PC system is designed.

The whole hardware structure of the system is shown in Fig. 11. The input of the system is a video stream from the high-speed camera – acA2000-340. 10 pixels are output simultaneously from the camera at a frequency of 82 MHz. When considering the limitation from the hardware, such as blanking interval, this data is transformed into 4 paralleled data with a frequency of 100 MHz through the camera link receiver. The 4 paralleled pixels are used as the input of the image process module. The data goes through Sobel, Harris, neighbouring check, descriptor generation and descriptor matching modules. Finally, a 32-bit data is outputted from the image process module. 16-bit data of the output includes image information, feature information, and matching information. The extra 16 bits can be used to store addi-
Fig. 12 Images used for sequence making. (a) Lena: 512×512 pixels; (b) Boat: 800×640 pixels.

Fig. 13 Sequences used for evaluation of rotation robustness.

Fig. 14 Datasets used for evaluation of other image conditions. (a) Bikes: 1000×700 pixels, blur change; (b) Trees: 1000×700 pixels, blur change; (c) Leuven: 921×614 pixels, illumination change; (d) UBC: 800×640 pixels, JPEG compression change; (e) Wall: 1000×700 pixels, viewpoint change.

Fig. 14 Details of these five image pairs.

3. Evaluation Results

3.1 Algorithm Evaluation

Algorithm evaluation is utilized to evaluate the validity of the intensity directed symmetry in the matching system. The development environment on software is Visual Studio C++ 2013. OpenCV 2.4.11 is utilized to finish the implementation on software.

3.1.1 Evaluation Datasets Introduction

In order to evaluate the validity of the intensity directed symmetry in rotation robustness, two sequences are made by the two images shown in Fig. 12. Figure 12 (a) is from the USC-SIPI image database†, and Fig. 12 (b) is from the Oxford dataset [19], [20]. The test sequences are shown in Fig. 13. For each test image, eight rotated images can be got from rotating the original test image every 11.25°. These 8 rotated images and the original image together constitute a test sequence. The original image and every rotated image can constitute an image pair. Each test sequence includes 8 image pairs, and totally 16 image pairs are used to evaluate the rotation robustness of the proposed algorithm.

Five image pairs are used to test if the proposed algorithm has any influence on other image conditions except the rotation, such as blur changes, illumination changes, JPEG compression, and viewpoint changes. These image pairs are from the Oxford dataset [19], [20], which is widely used and public available. Figure 14 shows the detail of these five image pairs.

### Footnotes

†http://sipi.usc.edu/database/
3.1.3 Experiment Results and Discussions

In order to check the validity of the proposed keypoint orientation calculation method, the matching performance of the conventional method (intensity centroid [17]) and the proposed method (intensity directed symmetry) are evaluated in the same matching system (excepts the orientation calculation part).

Table 1 shows the matching performance in rotation. Column 3 shows the matching performance of the proposed method. The matching scores of the proposed method for all the image pairs are all more than 70 %, which verifies the validity of the proposed method in rotation change. Column 4 shows the difference of the matching score between the conventional method and the proposed method. The difference of the matching score between these two methods are quite small (within 1.0 %), which implies that the proposed method doesn’t cause large influence in other image conditions, and the proposed method can achieve almost the same performance with the conventional method.

3.2 Hardware Evaluation

In the hardware evaluation, Kintex-7 (XC7K325T) as FPGA offered by Xilinx, is used for hardware evaluation. The logic synthesis on FPGA is performed by ISE 14.7.

Table 3 shows the hardware performance of the designed image process core. The scope of image process core in the whole hardware structure is shown in Fig. 11. Row 2–6 shows the resource utilization of the image process core. Currently, 4 pixels are processed simultaneously because of the limitation from the camera and the block size of the local maximum neighbouring check. With the pixel selection in the descriptor generation and descriptor matching part, there will be a relatively small resource increase when more pixels are processed simultaneously. And the increase will mainly happen in the keypoint detection part. Row 7–9 shows the hardware performance of the orientation calculation module. The process time for 1 frame is less than 1 ms. The image process core can work at a maximum frequency of 171.199 MHz. The design of the image process core supports to process 640 × 480 video at a frame rate of 1306 fps.

Table 4 shows the hardware performance of the orientation calculation module.
Fig. 15 Examples of the matching result using the image process core.

The resource usage of the proposed orientation calculation operator is quite small, and the DSP48E1 is completely removed by the proposed method. DSP48E1 is a digital signal processing logic element on FPGA devices. DSP48E1 makes it possible to perform different kinds of arithmetic operations on FPGA and is a precious resource in FPGA. Less use of DSP48E1 makes it possible to provide enough DSP48E1s for the further complicated design. Row 7–8 shows the performance of the orientation calculation module. The designed orientation calculation module can work at a maximum frequency of 541.62 MHz, and the process time of the module in the matching system is 0.0003 ms/frame.

4. Conclusions

In this paper, an FPGA based high frame rate and ultra-low delay matching system, which is robust to rotation, is put forward. Totally two methods are proposed to implement this system. Firstly, pixel selection based 4-1-4 parallel matching is proposed to design a parallel structure in the basic matching system. The pixel selection based 4-1-4 structure makes it possible to process four pixels at one time while using 1-thread resource consumption in the descriptor generation and descriptor matching part. Register storage based parallel matching accelerates the matching process by fully parallel the process in the descriptor matching part. Secondly, intensity directed symmetry is proposed to calculate the keypoint orientation in a hardware-friendly way from the algorithm level, and its appropriate hardware implementation is presented. Experiment results show that the proposed keypoint orientation operator achieves almost the same performance with the state-of-art intensity centroid orientation operator in the matching system, and the designed image process core supports to process VGA (640×480) videos at a process speed of 1306 fps and with a delay of 0.8083 ms/frame.

Our future research will dive more in the following aspects. In the technical aspect, the robustness of the matching system in other conditions, such as scale change and deformation, is added into consideration for further applications. In the aspect of applications, to provide the techniques needed in the human-machine interactive applications, the high frame rate and ultra-low delay matching system is considered to be applied into computer vision techniques such as object tracking, action recognition, and camera localization.

Acknowledgments

This work was supported by KAKENHI (26280016) and Waseda University Grant for Special Research Projects (2017B-261).

References

[1] W. Hu, W. Li, X. Zhang, S. Maybank, “Single and multiple object tracking using a multi-feature joint sparse representation,” IEEE Trans. Pattern Anal. Mach. Intell., vol.37, no.4, pp.816–833, 2015.
[2] Y. Guo, H. Zeng, Z.-C. Mu, F. Zhang, “Rotation-invariant DAISY descriptor for keypoint matching and its application in 3D reconstruction,” International Conference on Signal Processing, pp.1198–1201, 2010.
[3] R. Mur-Artal, J.M.M. Montiel, J.D. Tardos, “ORB-SLAM: A versatile and accurate monocular SLAM system,” IEEE Trans. Robot., vol.31, no.5, pp.1147–1163, 2015.

[4] J. Chen, T. Yamamoto, T. Aoyama, T. Takaki, and I. Ishii, “Simultaneous projection mapping using high frame-rate depth vision,” IEEE International Conference on Robotics and Automation, pp.4506–4511, 2014.

[5] J. Kelsch, M. Baumann, A. Schieben, T. Hesse, C. Loper, M. Dziennus, M. Heesen, and C. Brunken, “Interaction design of automatic steering for collision avoidance: Challenges and potentials of driver decoupling,” IET Intelligent Transport Systems, vol.9, no.1, pp.95–104, 2015.

[6] E. Rosten, R. Porter, and T. Drummond, “Faster and better: A machine learning approach to corner detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol.32, no.1, pp.105–119, 2010.

[7] C. Harris and M. Stephens, “A combined corner and edge detector,” Alvey Vision Conference, pp.147–151, 1988.

[8] J. Heinly, E. Dunn, and J.-M. Frahm, “Comparative evaluation of binary features,” European Conference on Computer Vision, vol.7573, pp.759–773, 2012.

[9] D.G. Lowe, “Distinctive image features from scale-Invariant key points,” Int. J. Comput. Vision., vol.60, no.2, pp.91–110, 2004.

[10] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded-up robust features,” European Conference on Computer Vision, vol.3951, pp.404–417, 2006.

[11] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: binary robust independent elementary features,” European Conference on Computer Vision, vol.6314, pp.778–792, 2010.

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” IEEE International Conference on Computer Vision, pp.2564–2571, 2011.

[13] T. Suzuki and T. Ikenaga, “Low complexity keypoint extraction based on SIFT descriptor and its hardware implementation for Full-HD 60fps video,” IEICE Trans. Fundamentals, vol.E96-A, no.6, pp.1376–1383, 2013.

[14] T. Rao and T. Ikenaga, “Quadrant segmentation and ring-like searching based FPGA implementation of ORB matching system for Full-HD video,” IAPR International Conference on Machine Vision Applications, pp.89–92, 2017.

[15] J. Weberrus, L. Kleeman, and T. Drummond, “ORB feature extraction and matching in hardware,” Australasian Conference on Robotics and Automation, pp.2–4, 2015.

[16] H. Heo, J.-Y. Lee, K.-Y. Lee, and C.-H. Lee, “FPGA based implementation of FAST and BRIEF algorithm for object recognition,” IEEE International Conference of IEEE Region 10, pp.1–4, 2013.

[17] P.L. Rosin, “Measuring corner properties,” Comput. Vis. Image. Und., vol.73, no.2, pp.291–307, 1999.

[18] T. Hu, H. Wu, and T. Ikenaga, “FPGA implementation of high frame rate and ultra-low delay tracking with local-search based block matching,” International Conference on Machine Vision and Information Technology, pp.93–98, 2017.

[19] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L.V. Gool, “A comparison of affine region detectors,” Int. J. Comput. Vision., vol.65, no.1-2, pp.43–72, 2005.

[20] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” IEEE Trans. Pattern Anal. Mach. Intell., vol.27, no.10, pp.1615–1630, 2005.