Temporally Forward Nonlinear Scale Space for High Frame Rate and Ultra-Low Delay A-KAZE Matching System

Songlin DU††††,†, Yuan LI††††, Nonmembers, and Takeshi IKENAGA††††, Member

SUMMARY High frame rate and ultra-low delay are the most essential requirements for building excellent human-machine-interaction systems. As a state-of-the-art local keypoint detection and feature extraction algorithm, A-KAZE shows high accuracy and robustness. Nonlinear scale space is one of the most important modules in A-KAZE, but it not only has at least one frame delay and but also is not hardware friendly. This paper proposes a hardware oriented nonlinear scale space for high frame rate and ultra-low delay A-KAZE matching system. In the proposed matching system, one part of nonlinear scale space is temporally forward and calculated in the previous frame (proposal #1), so that the processing delay is reduced to be less than 1 ms. To improve the matching accuracy affected by proposal #1, pre-adjustment of nonlinear scale (proposal #2) is proposed. Previous two frames are used to do motion estimation to predict the motion vector between previous frame and current frame. For further improvement of matching accuracy, pixel-level pre-adjustment (proposal #3) is proposed. The pre-adjustment changes from block-level to pixel-level, each pixel is assigned an unique motion vector. Experimental results prove that the proposed matching system shows average matching accuracy higher than 95%, which is 5.88% higher than the existing high frame rate and ultra-low delay matching system. As for hardware performance, the proposed matching system processes VGA videos (640 × 480 pixels/frame) at the speed of 784 frame/second (fps) with a delay of 0.978 ms/frame. 

key words: high frame rate, ultra-low delay, A-KAZE matching, temporally forward nonlinear scale space

1. Introduction

High frame rate and ultra-low delay are the most essential requirements in building excellent computer vision based human machine interactions, such as projection mapping [1], simultaneous localization and mapping (SLAM) [2], and automatic driving [3]. Currently, on the other hand, a common video is about 60 frames/second which is much lower than the requirement of high frame rate; on the other hand, the processing delay of most of the vision systems are relatively long, such as 0.1 second/frame or 0.01 second/frame, which also cannot satisfy the requirement of ultra-low delay. An important existing work which also aims at building a high frame rate and ultra-low delay matching system was proposed by Hu and Ikenage [4]. As for this work [4], although frame rate is high enough and processing delay is also low, its robustness is not high enough, because it uses several fixed template to handle objects’ scale change. Hu’s work [4] first detects keypoints in each video frame, and then matches the keypoints detected in each video frame with the keypoints detected on some templates prepared in advance. The keypoint detection module is based on FAST corner detection [5], and the matching algorithm is based on a famous local feature descriptor named oriented FAST and rotated BRIEF (ORB) [6], because FAST and ORB are efficient and hardware friendly. With FPGA based hardware implementation, the image process core of Hu’s work [4] processes VGA resolution (640 × 480) video sequences faster than 1000 fps, and the processing delay is 0.8083 ms/frame. The processing speed and delay satisfy the requirements of high frame rate and ultra-low delay, but its robustness is not high enough because it uses three size-fixed templates to do matching in order to solve the problem of objects’ scale change. For example, when scale of the target object greatly changes, the matching accuracy is dynamically decreased.

In order to improve the matching performance, especially for scale change, this paper focuses on designing a high frame rate and ultra-low delay nonlinear scale space based on accelerated KAZE (A-KAZE) algorithm [7]. A-KAZE algorithm [7] is a state-of-the-art matching algorithm which generates nonlinear scale space to acquire more scale-invariant features. Compared with other local feature extraction algorithms, such as SIFT [8] and SURF [9], nonlinear scale space is able to keep more boundary information of the original image, leading to higher matching accuracy. Meanwhile, since the descriptor of A-KAZE algorithm is binary, it is much faster than other algorithms. Because of the above reasons, A-KAZE algorithm is more suitable for a high accuracy, high frame rate, and ultra-low delay matching system. The process of A-KAZE algorithm is that for each input image, it generates nonlinear scale space firstly, then detects keypoints and generates descriptor for each keypoint, and finally uses these descriptors to do matching. Nonlinear scale space generation is the most important module in the whole A-KAZE algorithm, because this part influences the matching accuracy mostly. However, to achieve high frame rate and ultra-low delay with hardware implementation, some problems need to be solved. First, nonlinear scale space generation needs to do complex iterations for many
times, as the input of each octave is the output of the previous octave. So each octave needs to wait for the finish of previous octave. The delay of such kind of processing is obviously more than one frame, while high frame rate and ultra-low delay matching system requires to finish the processing of each frame within one frame delay. This is a serious problem preventing the realization of high frame rate and ultra-low delay matching system. Second, for each sublevel, nonlinear diffusion needs to be implemented. It is widely known that in the nonlinear diffusion equation, there are derivatives and divisions which are all not hardware friendly. Third, A-KAZE algorithm adopts unfixed number of iterations to approximate the results of nonlinear diffusion equation step by step. The unfixed number of iterations is not able to be directly implemented in hardware. What’s more, each sublevel uses previous sublevel’s results to do nonlinear diffusion, data dependency also exists between octaves. As a summary, because long time delay, complex calculations, unfixed number of iterations and data dependency exist, the original nonlinear scale space of A-KAZE is difficult to be implemented for a high frame rate and ultra-low delay matching system.

This paper proposes temporally forward nonlinear scale space for high frame rate and ultra-low delay A-KAZE matching system. In the proposed system, to remove complex calculations and unfixed number of iterations, the HFD algorithm [10] is utilized to replace nonlinear diffusion equation. All the calculations of HFD algorithm just include addition, subtraction, multiply and bit-width displacement. Furthermore, HFD algorithm does not require unknown times of iterations. To solve the problem of data dependency between octaves and to meet the requirement of ultra-low delay for the high frame rate video, the structure named temporally forward nonlinear scale space (proposal #1) is proposed. The main idea of this proposal is that one part has been processed in the previous frame. What’s more, to improve the matching accuracy, pre-adjustment of nonlinear scale space (proposal #2) and pixel-level pre-adjustment (proposal #3) are proposed. The relations among the three proposals are as follows. Proposal #1 is the top level structure designed for the purpose of high frame rate and ultra-low delay. Although proposal #1 makes the goal of high frame rate and ultra-low delay possible to be achieved, the matching accuracy is decreased. Proposal #2 is designed to recall the matching accuracy decreased by proposal #1. Proposal #3 further improves proposal #2 to make the matching accuracy to be higher. The work described in this paper is an extension of our previous conference paper [11], with significant new contents including both new proposals and new experimental results.

The rest of this paper is organized as follows. Section 2 presents the three proposals one by one. Section 3 presents the hardware structure of the proposed high frame rate and ultra-low delay system. Experimental results on both of software and hardware are reported and analysed in Sect. 4, followed by conclusion and future works given in Sect. 5.

2. Proposed A-KAZE Matching System

The framework of the whole proposed matching system is shown in Fig. 1. For each input image to do matching, the proposed system builds nonlinear scale space firstly, and then performs keypoint detection and descriptor generation. The basic matching system which just contains keypoint detection step and descriptor generation step was designed in conventional work [4], [12]. Harris Corner Detector [13] is used to detect the corners of the input image as keypoints. For all these keypoints, binary descriptors are generated by binary robust independent elementary features (BRIEF) [14]. The obtained descriptors are used to do matching. The matching mechanism is brute-force matching measured by Hamming distance, i.e. if the Hamming distance is smaller than threshold, the corresponding descriptors are matched, otherwise they are not matched.

Keypoint detection is to identify locations that are invariant to scale change. This can be achieved by searching for stable keypoints across all possible scales, using a continuous function. All the possible scales constitute a scale space. Following the definitions in SIFT [8], a scale space is composed of several octaves, and an octave is composed of several sublevels. Different algorithms use different schemes to construct octaves and sublevels for building scale space. For example, in the well-known SIFT [8], for each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce a set of scale space images. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated. In A-KAZE algorithm [7], the octaves and sublevels are obtained through a nonlinear way which does not perform down-sampling at each new octave.

For each frame, nonlinear scale space contains two frame-level processes. In the condition of original A-KAZE algorithm’s nonlinear scale space structure, there exists data dependency between the first frame-level process and the second frame-level process. In particular, each frame-level
process needs to wait for the finish of the previous frame-level process. As a result, the delay of this structure is very long, so it is unable to meet the requirements of ultra-low delay. However, for the proposed structure of nonlinear scale space, the second frame-level process of current frame uses the results of the first process of previous frame and just with some pre-adjustments to keep the matching accuracy. It is obviously to know that the speed of nonlinear scale space of the proposed structure is twice faster than the original algorithm. The process time for each frame of the proposed structure is less than one frame time. It therefore meets the requirements of high frame rate and ultra-low delay matching system. It is worth noting that the proposed concept not only can be applied for A-KAZE algorithm's nonlinear scale space to implement in hardware and achieve ultra-low processing delay, but also can be applied for all this kind of algorithms which need several frame-level processes caused by data dependency.

In the step of nonlinear scale space generation, HFD[10] algorithm is utilized. In the proposed structure, octave 0 of current frame is generated in the step of nonlinear scale space, and this octave 0’s information will be used. Through the process of the pre-adjustment, this octave is utilized as the octave 1 of next frame. In the process of pre-adjustment, pixel-level pre-adjustment is adopted. The structure of finishing a part of current nonlinear scale space in previous frame is proposal #1, which temporally forwards nonlinear scale space. Doing some adjustment on previous octave 0 to obtain current octave 1 is proposal #2, i.e. pre-adjustment of nonlinear scale space. What’s more, to further improve the matching accuracy of pre-adjustment, proposal #3, i.e. pixel-level pre-adjustment is proposed at the same step.

Furthermore, to make clear the parallelism of the whole proposed structure, it is needed to be explained with more details. Octave 1 of current frame can be obtained before finishing the process of the octave 0. And the pre-adjustment for the octave 1 of the next frame is parallel with the step of keypoint detection of current frame as well. As a result, there is no extra delay of pre-adjustment step. The proposed structure is able to save a lot of time compared with the original A-KAZE algorithm’s nonlinear scale space structure.

2.1 Proposal #1: Temporally Forward Nonlinear Scale Space

Proposal #1 completely changes the original structure of the nonlinear scale space generation to meet the requirements of high frame rate and ultra-low delay. Proposal #1 utilizes the property of high frame rate video that temporal coherence is strong between continues frames. Because the frame rate of input video reaches 784 fps, the differences between each adjacent frames are small. The similarity between each adjacent frames is high even for object translation or rotation. Because of the high similarity between adjacent frames, the previous frame can be used as an approximation of the next frame, without decreasing much accuracy.

The comparisons of the original structure and the proposed structure is shown in Fig. 2. In the original structure, octave 0 and octave 1 are processed sequentially and both of them need one frame time, while in the proposed structure, octave 0 and octave 1 are processed in parallel. Also, keypoint and descriptor parts are parallel with them, the whole process of nonlinear scale space finishes in one frame time. For the original structure, both octave 0 and octave 1 are generated using the information of current frame k. But in our proposal #1, only octave 0 is generated by frame k. The octave 1 of frame k is generated by the information of frame k − 1. Because there is no data dependency between the proposed nonlinear scale space’s octave 0 and octave 1, octave 1 has no necessity to wait for the processing of octave 0. As a result, problem of the conventional work is solved and delay is decreased to be less than one frame time.

The detailed process of proposal #1 are as follows. To achieve high robustness to scale change, the mechanism of nonlinear scale space requires each octave to be smaller than the previous one. To this end, the process of proposal #1 is that octave #1 of current frame is the downsampled octave #0 of previous frame, and octave #0 of current frame need to be generated. The downsampling is calculated through averaging the pixels in within a local region. Formally, for s times of down sampling, the result value is

$$p_k = \sum_{i \in \text{win}(k)} \frac{I_i}{s^2},$$

(1)

where $p_k$ is the downsampled result, $I_i$ denotes the pixel intensity in the local region $\text{win}(k)$ of size $s^2$. 

![Fig. 2](image-url) Conceptual difference between conventional work and proposal #1.
2.2 Proposal #2: Pre-Adjustment of Nonlinear Scale Space

Pre-adjustment of nonlinear scale space is achieved through motion estimation. Motion estimation [15] is a general concept widely used in many computer vision related fields, such as video compression and object tracking. The basic process of motion estimation is shown in Fig. 3. Since there exists correlation between the adjacent frames, the best match can be found in the search area in the reference frame. The position differences between the current block and the best match block are the motion vector. The process of obtaining a motion vector is called motion estimation. In this section, the blocks for motion estimation are defined as non-overlapping grids of 16 × 16 pixels in the image.

2.2.1 Selective Gray-Coded Bit-Plane Based Low-Complexity Motion Estimation

There are a lot of kinds of motion estimation algorithms. From many motion estimation algorithms, selective gray-coded bit-plane based low-complexity motion estimation [16] stands out because of its high processing speed and good performance. The processing speed of this motion estimation method meets the requirements of high frame rate and ultra-low delay matching system. This algorithm firstly generates gray-coded bit-planes [17] of the input image. The $K$-bit gray code of a pixel value is computed by [17]

\[ g_{k-1} = a_{k-1}, \]

\[ g_k = a_k \oplus a_{k+1}, 0 \leq k \leq K - 2, \]

where $a$ means the binary code of a pixel value, $g$ means the gray code of this pixel value, “⊕” denotes XOR operation. The meaning of gray-coded bit-plane is as follow: $g_0$ means that the pixel value of this bit-plane is just the $k$-th position value of the whole gray code of this pixel. The reason why choose gray-coded bit-plane instead of binary coded bit-plane is that gray code is robust as successive gray code words differ in only one bit position.

Secondly, this motion estimation algorithm chooses three bit-planes of highest position, because they are considered to contain the most significant information of the input image. The higher position is, the more important the binary code is. Because binary code of higher position has stronger influence on the pixel value than lower ones. At the same time, it will be less detailed. For example, if $a_7$ changes from 1 to 0, the pixel value will be 125 lower. But if $a_0$ changes from 1 to 0, the pixel value will only be 1 lower. The importance of position in gray code is similar with the binary code. The 8-bit gray-coded bit-planes are shown in Fig. 4. Through the analysis of gray coded bit-planes, it can be found that although $g_7$ takes the most important position, it does not provide any motion information, as is just a large white or black area. What’s more, the lowest four bit-planes include too much detailed information, the background does not influence the object too much to obtain accurate motion information. As a result, three lower position gray coded bit-planes, i.e. $g_6$, $g_5$ and $g_4$, which obtain accurate motion information are chosen to do motion estimation in the proposed system.

After obtaining three gray-coded bit-planes of the input image, XOR calculations is performed to find the most similar motion for motion estimation. The equation of matching criterion (MC) is [17]

\[
MC (m, n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \sum_{k=NTB}^{K-1} 2^{k-NTB} \times \left\{ g_k^c(i, j) \oplus g_k^r(i + m, j + n) \right\}, \tag{4}
\]

where $NTB$ denotes the number of truncated bits. When $NTB = 5$, it means that only the three most important bit-planes are utilized in the matching process. $m$ and $n$ mean the position of this pixel. $N$ is the size of search window. $g_k^c$ is the gray-coded bit-plane in the current frame, while $g_k^r$ is the gray-coded bit-plane in the reference frame. Because of all these calculations are very simple, the processing speed of this motion estimation algorithm is very fast and hardware friendly.

2.2.2 Process of Pre-Adjustment

Because of the differences between the previous frame and the current frame, if only proposal #1 is utilized, the matching performance of the whole proposed matching system decreases, although the aim of proposal #1 is not to achieve
To solve this problem, the purpose of proposal #2, i.e. pre-adjustment of nonlinear scale space, is to heighten the similarity between the octave 1 calculated by previous frame and the octave 1 of current frame. And then, the matching accuracy is raised. At the same time, to meet the requirements of high frame rate and ultra-low delay matching system, the process of adjustment should be finished before the current frame comes. Pre-adjustment is therefore proposed. Previous two frame’s information are utilized to do motion estimation to predict the motion vector from previous frame to current frame. Adding the motion vector to the calculated octave 1, the predicted octave 1 of current frame is obtained.

Selective gray-coded bit-plane based low-complexity motion estimation is used to do motion estimation and get the motion vector. The comparisons between the original octave 1 of proposed nonlinear scale space and adding pre-adjustment of nonlinear scale space are shown in Fig. 5. If just proposal #1 is utilized, the information of frame \( k_1 \) is directly used to do downsampling and get the octave 1 of frame \( k \). However, after adding the pre-adjustment, the octave 1 of frame \( k \) is obtained by getting the motion vector through the motion estimation calculated by frame \( k_1 \) and frame \( k_2 \). And then using this motion vector to do motion prediction to predict the motion between frame \( k \) and frame \( k_2 \). Predicted octave 1 of frame \( k \) is obtained by adding the predicted motion to downsampled octave 0 of frame \( k_1 \).

The coordinate of the pixel in predicted octave 1 (\( x_{\text{prediction}}, y_{\text{prediction}} \)) is calculated by

\[
(x_{\text{prediction}}, y_{\text{prediction}}) = (x_{\text{ori}} + x_{\text{motion}}, y_{\text{ori}} + y_{\text{motion}}),
\]

where \( x_{\text{ori}} \) and \( y_{\text{ori}} \) denote the original coordinates, and \( x_{\text{motion}} \) and \( y_{\text{motion}} \) denote the motion vectors. For border problem, replication is used when the situation that pixel moves to other place and there is no pixel in this position happens.

2.3 Proposal #3: Pixel-Level Pre-Adjustment

For the original motion estimation method utilized in the pre-adjustment of proposal #2, the whole block is used to calculate motion vector. And then, resulting in that each pixel in this block has the same motion vector, they are moved in the same direction. But this method has a problem, as shown in Fig. 6. The motion vectors for pixels which are in the same block have different motions in real-world situations, especially in the situation of rotation. As for the weakness of proposal #2, because different pixels are assigned the same motion vector and background pixels move as the same as object pixels, the accuracy of original proposal #2 is not high enough. To solve these problems, proposal #3, i.e. pixel-level pre-adjustment, is proposed. In proposal #2, we follow the standard selective gray-coded bit-plane based motion estimation [16], so the blocks are non-overlapping. But in proposal #3, to estimate motion vector for every pixel, the blocks are overlapped.

The comparisons between the original pre-adjustment method and pixel-level pre-adjustment are shown in Fig. 7. For the original method, the whole block pixels are used to do motion estimation and the obtained motion vector is distributed to all the pixels in this block. In the situation of proposal #3, the motion vector got by calculating the whole block’s pixels is just assigned to the pixel in the center of this block. So, each pixel has its own independent pre-adjustment result. The pre-adjustment becomes more pre-

![Fig. 5](image-url) Conceptual difference between conventional work and proposal #2.

![Fig. 6](image-url) Problem of conventional pre-adjustment.

![Fig. 7](image-url) Conceptual difference between conventional work and proposal #3.

![Fig. 8](image-url) Detailed process of proposal #3.
Moreover, changing pre-adjustment method from the whole block to only the center pixel does not cause any additional delay to the matching system. Because the whole pre-adjustment part is processed in parallel with previous frame’s keypoint detection part. Also, due to the very simple calculations and high parallelism of pre-adjustment, the processing speed of this improved pre-adjustment method meets the requirements of high frame rate and ultra-low delay. Figure 8 shows the detailed process of proposal #3. The step size of original pre-adjustment is a block length, the calculation of motion estimation is block-by-block. However, for proposal #3, the step size of motion estimation is just a pixel, it is calculated pixel-by-pixel. As a result, each pixel has different and independent motion vector. The accuracy of pre-adjustment is therefore improved a lot.

3. Hardware Structure of the Matching System

High frame rate and ultra-low delay matching system consists of three parts including PC, high frame rate camera, and FPGA board, as shown in Fig. 10. Each image captured by high frame rate camera is transported to FPGA in form of pixel stream. In the FPGA board, matching algorithm is performed, and the matching results are transported to PC. Figure 9 shows the specific hardware structure of the proposed matching system. The image information captured by high frame rate camera is received and processed by the camera link receiver module. And then the pixel intensity information is transported to image processing core which contains modules of nonlinear scale space, pre-adjustment, keypoint detection, descriptor generation and matching. The five main modules are connected with register access. What’s more, the register access is attached with USB 3.0 interface. Through this way, PC is able to communicate with FPGA to adjust relevant parameters and threshold values which are used in image processing core.

In nonlinear scale space module, octave 0 is calculated. And this module access to memory, the calculated octave 0 of current frame is saved in the memory to do pre-adjustment for octave 1 of the next frame (proposal #1). Also previous two frames are saved in the memory too to do the pixel-level motion estimation for the preadjustment (proposal #2 and proposal #3). The nonlinear scale space module and the pre-adjustment module are just connected with register access and memory access separately, and there is no data dependency between these two modules. But saving all data of nonlinear scale space module and pre-adjustment module in the memory of FPGA board causes the problem of memory insufficiency. Downsampling is therefore demanded to be implemented before saving data to memory. After that, the data saved in the memory is just one quarter. Via memory controller, the output matching results are written into double-data-rate three synchronous dynamic random access memory (DDR3-SDRAM). Output results are read by PC through DDR3-SDRAM.

4. Experimental Results and Analysis

4.1 Evaluation Environment

To evaluate the feasibility and matching accuracy of the proposals, experiments on both software and hardware are performed. The matching accuracy is evaluated through software experiments. The comparisons of proposals’ matching performance and previous work’s matching performance are evaluated in software as well. The software evaluation environment is Visual Studio 2013 with OpenCV 2.4.11 on a PC of Windows 10 professional operating system.

The experiments on hardware are used to evaluate the processing speed and delay to find out whether the designed matching system meets the requirements of high frame rate.
and ultra-low delay or not. At the same time, hardware resource utilization is also evaluated to find out whether the proposals are feasible to be implemented for real applications. Hardware environment consist of three parts as shown in Fig. 12, i.e. Xilinx Kintex-7 XC7K325T FPGA board, BASLER acA2000-340 high frame rate camera, and a PC of Core i7-4790 3.6GHz CPU. The general specifications of FPGA board is shown in Table 1. What’s more, logic synthesis and implementation of FPGA board are performed by Vivado 2017.2.

The dataset which is utilized to evaluate the matching accuracy of the proposed matching system contains four kinds of test sequences. These test sequences are all captured by high frame rate camera. The frame rate is 784 fps. What’s more, the resolution for each frame in these sequences is 640 × 480 pixels. There are 1200 frames in each sequence. The evaluation dataset contains representative situations, such as translation, rotation, illumination change, and scale change. The typical frames of these sequences are shown in Fig. 11. Translation in Fig. 11(a) means the object moves from top to bottom. In the situation of rotation, the object is in the center of the images and rotates in the plane. Illumination change means that the object does not move but the lighting condition is changed, such as from dark to bright. For scale change, the size of the object is changed by moving towards the camera from a distance. Before matching, the keypoints detected in the two frames are manually observed to count the number of ground-truth matches. After matching two frames by our proposals and other comparison algorithms, the keypoints in the two frames are connected. By displaying the connections one-by-one, all the matches are manually observed to distinguish and to count correct ones and false ones.

The widely used metric F-score [18] is adopted to evaluate the matching accuracy of the designed high frame rate and ultra-low delay matching system. The definition of F-score is based on two fundamental metrics Precision and Recall, i.e.

\[
\text{Precision} = \frac{\# \text{correct matches}}{\# \text{correct matches} + \# \text{false matches}} \tag{6}
\]

and

\[
\text{Recall} = \frac{\# \text{correct matches}}{\# \text{total matches in ground truth}} \tag{7}
\]

where Precision is defined as the percentage of correct matches in all the matches. Higher precision means that the system makes less mistakes in matching correspondence keypoints. Recall is defined as the percentage of correct matches in ground-truth matches. Higher Recall means that the system losses less correspondence keypoints. Considering Precision and Recall simultaneously, F-score is defined as

\[
\text{F-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}
\]

Higher F-score means that the system obtains more true
matches and lost less true matches. Therefore, higher F-score indicates higher performance.

4.2 Software Evaluation Results

The matching performances of proposals, original A-KAZE[7] and existing high frame rate and ultra-low delay matching system [4], [12] are evaluated on four test sequences. The evaluation results are reported in Table 2. Proposal #1 is that just change the structure of nonlinear scale space to make it temporally forward. Proposals #1 & #2 mean that pre-adjustment is added to proposals #1. Proposals #1 & #2 & #3 refer to the whole proposed matching system with preadjustment is changed from block-level to pixel-level. The algorithms of keypoint detection and descriptor generation part of all the proposed matching system are the same with existing matching system.

Firstly, compared with the existing matching system, original A-KAZE algorithm shows a much better performance for all cases. The average F-score of A-KAZE algorithm is 7.69% higher than the existing matching system. When proposal #1 is implemented only, i.e. just uses previous frame’s information to calculate the octave 1 of current frame, the matching accuracy decreases a lot, especially in the situation of rotation. The average F-score of proposal #1 is 4.88% lower than original A-KAZE. Furthermore, in the case of rotation, it is 7.90% lower than original A-KAZE algorithm. When adding proposal #2, average F-score increases about 1%. But the matching performance for the situation of rotation is still low, as it is still 89.98%. To further improve the matching accuracy, especially for rotation, proposal #3 is added. The evaluation results show that, after adding proposal #3 to the designed nonlinear scale space, the matching performance improves a lot in all cases. Finally, the matching accuracy of the whole designed system is 95.58%. The matching accuracy for rotation and scale change become 5.76% and 6.59% higher than the existing matching system [4].

Meanwhile, compared with the original A-KAZE algorithm, the proposed matching system’s matching accuracy for the situation of rotation is still a little low. There are two main reasons. First, the descriptor generated by BRIEF algorithm does not have high robustness for rotation. Second, the motion vectors between two frames have relatively-large differences, using previous two frames to do motion estimation to predict the motion of current frame leads to the low robustness for rotation. To solve these problems, on one hand, it is possible to use more rotation-robust descriptors; on the other hand, more accurate motion estimation methods can be introduced to further improve the robustness to rotation.

4.3 Hardware Evaluation Results

The proposed high frame rate and ultra-low delay matching system is implemented on FPGA to evaluate this system’s feasibility in real applications. Table 3 reports the utilization of hardware resources by the proposed matching system. One can conclude from Table 3 that most types of the utilized hardware resources are less than half of their total amount on the FPGA board, which means the implemented high frame rate and ultra-low delay A-KAZE matching system is resource-saving. One can also find that the utilization of LUT is 96%, because generating sublevels and octaves need to save much data in the RAM. What’s more, because HFD algorithm needs to do great quantity of calculations whose input is a 7 × 7 matrix. There are several potential solutions to reduce the utilization of LUT. Currently, two same sized RAMs are used to save one image to decrease delay, if this structure changes from parallel to sequential, using single port RAM to replace dual port RAM, the memory used to save the data can be a half of the current one. Of course, the processing delay will have one more clock delay because of the time consuming of reading and writing. Also, there are many intermediate data needs to be saved in the calculation of HFD algorithm, finding other simpler nonlinear diffusion filter algorithm is possible to save more LUTs.

Table 4 reports the hardware performance of the proposed A-KAZE matching system. The input frequency

| Table 2 | Comparisons on F-score (%) among five algorithms. |
|---------|-----------------------------------------------|
| Translation | Existing matching system [4] | A-KAZE [7] | Proposal #1 | Proposals #1 & #2 | Proposals #1 & #2 & #3 |
| Rotation | 89.96 | 96.23 | 93.89 | 94.88 | 96.18 |
| Illumination change | 86.94 | 97.20 | 89.30 | 89.98 | 92.70 |
| Scale change | 92.30 | 98.68 | 93.51 | 94.24 | 97.23 |
| Average | 89.70 | 97.39 | 92.51 | 93.45 | 95.58 |

| Table 3 | Hardware resource utilization by the proposed A-KAZE matching system. |
|---------|-----------------------------------------------|
| Resource | Utilized amount | Percentage of utilization |
| LUT | 196134 | 96% |
| LUTRAM | 28068 | 44% |
| FF | 157122 | 39% |
| BRAM | 297 | 65% |
| DSP | 228 | 27% |
| IO | 208 | 42% |
| BUFG | 13 | 41% |
| MMCM | 4 | 40% |

| Table 4 | Hardware performance of the proposed A-KAZE matching system. |
|---------|-----------------------------------------------|
| Input frequency | 100.00 MHz |
| Input frame rate | 784 fps |
| Processing delay | 0.978 ms/frame |
and the input frame rate are 100.00 MHz and 784 fps, respectively. The processing delay of the proposed A-KAZE matching system is 0.978 ms/frame. Since the delay is less than 1 ms/frame, it satisfies the requirement of ultra-low delay. As a summary, the proposed matching system processes video of 784 fps with the delay of 0.978 ms/frame, it successfully achieves the goal of high frame rate and ultra-low delay.

5. Conclusion and Future Works

A temporally forward nonlinear scale space for high frame rate and ultra-low delay A-KAZE matching system has been proposed in this paper. In the proposed matching system, one part of nonlinear scale space is temporally forwarded and calculated in the previous frame, so that the processing delay is reduced to be less than 1 ms. To improve the matching accuracy affected by temporally forwarding, previous two frames are used to do motion estimation to predict the motion vector between previous frame and current frame. For further improvement of matching accuracy, pixel-level preadjustment is proposed. The pre-adjustment changes from block-level to pixel-level, each pixel is assigned an unique motion vector. Experimental results show that the proposed matching system achieves high matching accuracy and processes VGA videos at the speed of 784 fps with a delay of 0.978 ms/frame.

In future works, more rotation-robust descriptors and more precise motion estimation methods will be investigated to further improve the robustness of the proposed high frame rate and ultra-low delay matching system. In particular, proposal #3 straightforwardly extending block-level motion estimation to pixel-level motion estimation, which makes a balance between speed and robustness. In our future work, it is an interesting direction to investigate how to further improve the robustness of motion estimation with simple calculations and hardware-friendly operations.

Acknowledgments

This work was supported by KAKENHI (16K13006) and Waseda University Grant for Special Research Projects (2019C-581).

References

[1] E.L. Schwartz, “Spatial mapping in the primate sensory projection: analytic structure and relevance to perception,” Biological Cybernetics, vol.25, no.4, pp.181–194, 1977.
[2] M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csonka, “A solution to the simultaneous localization and map building (SLAM) problem,” IEEE Trans. Robotics and Automation, vol.17, no.3, pp.229–241, June 2001.
[3] M. Heesen, M. Dziennius, T. Hesse, A. Schieben, C. Brunken, C. Löper, J. Kelsch, and M. Baumann, “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, 2014.
[4] T. Hu and T. Ikenage, “Pixel selection and intensity directed symmetry for high frame rate and ultra-low delay matching system,” IEICE Trans. Inf. & Syst., vol.E101-D, no.5, pp.1260–1269, May 2018.
[5] 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, Jan. 2008.
[6] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” Int. Conf. Comput. Vis., 2011.
[7] P. Alcantarilla, J. Nuevo, and A. Bartoli, “Fast explicit diffusion for accelerated features in nonlinear scale spaces,” British Machine Vision Conference, pp.13.1–13.11, 2013.
[8] D.G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vis., vol.60, no.2, pp.91–110, 2004.
[9] H. Bay, T. Tuytelaars, and L.V. Gool, “Surf: Speeded up robust features,” European Conference on Computer Vision, pp.404–417, 2006.
[10] H. Siddiqui, M. Boutin, and C.A. Bouman, “Hardware-friendly descreening,” IEEE Trans. Image Process., vol.19, no.3, pp.746–757, March 2010.
[11] Y. Li, S. Du, and T. Ikenaga, “Temporally forward nonlinear scale space with octave prediction for high frame rate and ultra-low delay A-KAZE matching system,” Int. Conf. Machine Vision Applications, 2019.
[12] T. Hu and T. Ikenage, “FPGA implementation of high frame rate and ultra-low delay vision system with local and global parallel based matching,” Int. Conf. Mach. Vis. Appli., pp.286–289, 2017.
[13] C. Harris and M. Stephens, “A combined corner and edge detector,” Alvey vision conference, 1988.
[14] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” European Conference on Computer Vision, pp.778–792, 2010.
[15] K.R. Rao and J.J. Hwang, Techniques and Standards for Image, Video, and Audio Coding, Prentice Hall, New Jersey, 1996.
[16] S. Yavuz, A. Celebi, M. Aslam, and O. Urhan, “Selective gray-coded bit-plane based low-complexity motion estimation and its hardware architecture,” IEEE Trans. Consum. Electron., vol.62, no.1, pp.76–84, Feb. 2016.
[17] B. Natarajan, V. Bhaskaran, and K. Konstantinides, “Low-complexity block-based motion estimation via one-bit transforms,” IEEE Trans. Circuits Syst. Video Technol., vol.7, no.4, pp.702–706, Aug. 1997.
[18] C. Gouet and E. Gaussier, “A probabilistic interpretation of precision, recall and F-score, with implication for evaluation,” European Conference on Information Retrieval, pp.345–359, 2005.

Songlin Du received the Ph.D. degree from the Graduate School of Information, Production and Systems, Waseda University, Kitakyushu, Japan. He is currently with the School of Automation, Southeast University, Nanjing, China. His research interests include visual feature representation and related hardware implementation. He received the Best Paper Award at IS-PACS2017. He is a member of the IEEE.
Yuan Li received the B.E. degree in Information Engineering from Southeast University, China, in 2018, and the M.E. degree from Graduate School of Information, Production and Systems, Waseda University, Japan, in 2019. Her research focuses on FPGA implementation of computer vision related algorithms.

Takeshi Ikenaga received his B.E. and M.E. degrees in electrical engineering and Ph.D. degree in information & computer science from Waseda University, Tokyo, Japan, in 1988, 1990, and 2002, respectively. He joined LSI Laboratories, Nippon Telegraph and Telephone Corporation (NTT) in 1990, where he had been undertaking research on the design and test methodologies for high performance ASICs, a real-time MPEG2 encoder chip set, and a highly parallel LSI & system design for image understanding processing. He is presently a professor in the system integration field of the Graduate School of Information, Production and Systems, Waseda University. His current interests are image and video processing systems, which covers video compression (e.g. H.265/HEVC, SHVC, SCC), video filter (e.g. super resolution, noise reduction, high-dynamic range imaging), and video recognition (e.g. sport analysis, feature point detection, object tracking). He is a senior member of the Institute of Electrical and Electronics Engineers (IEEE), a member of the Institute of Electronics, Information and Communication Engineers of Japan (IEICE) and the Information Processing Society of Japan (IPSJ).