IEICE TRANSACTIONS

on Information and Systems

DOI:10.1587/transinf.2023EDP7279

Publicized:2024/07/23

This advance publication article will be replaced by the finalized version after proofreading.

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

# Aggregated to Pipelined Structure Based Streaming SSN for 1-ms Superpixel Segmentation System in Factory Automation

**Yuan LI**[†a)], **Tingting HU**[††], **Ryuji FUCHIKAMI**[††], *Nonmembers*, *and* **Takeshi IKENAGA**[†], *Member*

**SUMMARY** 1 millisecond (1-ms) vision systems are gaining increasing attention in diverse fields like factory automation and robotics, as the ultra-low delay ensures seamless and timely responses. Superpixel segmentation is a pivotal preprocessing to reduce the number of image primitives for subsequent processing. Recently, there has been a growing emphasis on leveraging deep network-based algorithms to pursue superior performance and better integration into other deep network tasks. Superpixel Sampling Network (SSN) employs a deep network for feature generation and employs differentiable SLIC for superpixel generation. SSN achieves high performance with a small number of parameters. However, implementing SSN on FPGAs for ultra-low delay faces challenges due to the final layer's aggregation of intermediate results. To address this limitation, this paper proposes an aggregated to pipelined structure for FPGA implementation. The final layer is decomposed into individual final layers for each intermediate result. This architectural adjustment eliminates the need for memory to store intermediate results. Concurrently, the proposed structure leverages decomposed layers to facilitate a pipelined structure with pixel streaming input to achieve ultra-low latency. To cooperate with the pipelined structure, layer-partitioned memory architecture is proposed. Each final layer has dedicated memory for storing superpixel center information, allowing values to be read and calculated from memory without conflicts. Calculation results of each final layer are accumulated, and the result of each pixel is obtained as the stream reaches the last layer. Evaluation results demonstrate that boundary recall and under-segmentation error remain comparable to SSN, with an average label consistency improvement of 0.035 over SSN. From a hardware performance perspective, the proposed system processes 1000 FPS images with a delay of 0.947 ms/frame.

***key words:*** *1-ms vision system, superpixel segmentation, superpixel sampling network (SSN), real-time, FPGA*

## 1. Introduction

In recent years, machine vision, recognized as a prominent non-contact inspection technology, has found extensive applications across diverse domains. A typical machine vision system comprises a sensing camera, a machine vision algorithm for data processing, and feedback to the actuator. While some applications, like remote sensing and scientific research, do not demand low latency, their performance benefits from high-resolution images. Conversely, the imperative for ultra-low delay in factory automation (FA) and robotics stems from the critical necessity for seamless interactions and heightened operational efficiency. The real-world scenes change continuously during the algorithm processing. In factory automation scenarios, the assembly line remains in motion while algorithms are being processed. Lengthy processing delays lead to invalid feedback results since objects have already shifted positions. Halting the assembly line until processing is complete significantly reduces efficiency. Similarly, in robotic scenarios, emergencies may arise suddenly. For instance, if a cup is falling, prolonged processing times lead to significant scene changes, such as the cup hitting the ground and breaking. Implementing an ultra-low delay system ensures minimal changes in the real-world scene and enables a prompt response from the actuator. This approach guarantees sustained high efficiency and effective handling of dynamic real-world situations. As the actuators are able to work at the frequencies of 1kHz [1], systems capable of processing 1000 frames per second (FPS) with processing speeds under 1 ms are desired. Field-Programmable Gate Arrays (FPGAs) emerge as a prevalent choice for the realization of 1-ms vision systems, attributed to their stream-based architecture and parallelism. 1-ms vision systems have already been realized in various fields, such as template matching [2], object tracking [3] and line detection [4].

Superpixel segmentation is the over-segmentation of images through grouping pixels based on low-level image properties. This process serves to reduce the number of image primitives for subsequent processing while capturing object boundaries. Consequently, superpixel segmentation is widely applied in machine vision tasks, such as image classification [5] and stereo matching [6]. Currently, superpixel segmentation is approached through two primary methods: hand-crafted algorithms and deep network-based algorithms. Simple linear iterative clustering (SLIC) [7] characterizes each pixel by 5-dimensional positional and color features. Linear spectral clustering (LSC) [8] augments SLIC by projecting these 5-dimensional features to a 10-dimensional space by kernel functions. Li *et al.* [9], [10] achieve a 1-ms SLIC by separating iteration of SLIC into the temporal domain. However, driven by the pursuit of enhanced performance and integration into other deep network tasks, a growing number of researchers have embraced deep networks for superpixel segmentation. Superpixel sampling network (SSN) [11] is the first end-to-end trainable network architecture for superpixel segmentation. SSN leverages a deep network to generate learned features for each pixel and employs differentiable SLIC to generate superpixels. The introduction of task-specific reconstruction loss enables the

[†]The authors are with the Graduate School of Information, Production and Systems, Waseda University, Kitakyushu-shi, 808-0135 Japan.

[††]The authors are with Panasonic Connect Co., Ltd., Fukuoka-shi, 812-8531 Japan.

a) E-mail: liyuan3104@fuji.waseda.jp

optimization of superpixels for integration into other deep network tasks. SCN [12] directly applies the U-net architecture to predict pixel-to-superpixel associations. AINet [13] introduces an AI module to embed pixel-neighboring grid relations for pixel–superpixel association.

To accelerate deep network on FPGAs with the objective of achieving ultra-low delay, a hardwired type architecture has been proposed [14]. This design affords a high degree of parallelism by directly mapping the entire network onto the FPGA. Crucially, this architecture removes the limitations of memory access. Contrary to the direct mapping approach, storing weights, inputs, and outputs in the external memory of the FPGA presents challenges to improving the overall latency due to the limited memory bandwidth of the external memory. The removal of such memory access limitations enables the hardwired type architecture to attain ultra-low delay performance. However, the hardwired type architecture imposes strict constraints on the number of parameters and network size. While SCN and AINet exhibit superior performance, the U-net structure with skip connections results in an excess of two million parameters and imposes substantial computation and memory requirements. Their structure makes them hard to implement on FPGAs to reach ultra-low delay. SSN maintains high performance while constraining parameters under 0.2 million, it is adopted as the basic structure for a 1-ms deep network-based superpixel segmentation system. In addition to the hardwired type architecture, the entire system must be fully pipelined and handle the input pixel stream promptly. A fully pipelined structure helps mitigate delays caused by waiting between modules, while streaming processing ensures the entire system's delay from sensor to feedback remains within 1 ms. Implementing SSN on an FPGA requires addressing the challenges posed by the aggregated structure. The aggregated structure makes it impractical to fully pipeline the entire system due to prolonged processing time of the aggregated step. Storing intermediate results for the aggregated step poses challenges for FPGA implementation with limited memory resources. To achieve streaming processing, memory conflicts between each step must be addressed. To handle these problems, this paper proposes an aggregated to pipelined structure with its system-level hardware implementation. The contributions of this paper are summarized as follows:

1. Aggregated to pipelined structure for FPGA implementation is proposed. The final layer of SSN, originally designed for aggregating all intermediate results, is decomposed into discrete final layers corresponding to each intermediate result. With the proposed structure, the necessity to store intermediate results is eliminated and the entire system undergoes pipelining with a pixel streaming input.
2. Layer-partitioned memory architecture is proposed. This architecture allocates dedicated memory for superpixel center information to each final layer. Consequently, the outcomes of each final layer are acquired with pixel streaming, circumventing conflicts arising from operations.
3. The proposed architecture has been implemented on an FPGA to develop a 1-ms superpixel segmentation system. A series of comprehensive experiments have been conducted to comprehensively validate both the algorithmic and hardware performance of the system.

The subsequent sections of this paper are organized as follows. Section 2 provides an overview of related works. Section 3 shows the proposed methods and implementation details. Experimental results and their analysis are presented in Section 4. Finally, Section 5 makes a conclusion.

## 2. Related Works

### 2.1 Deep Network-Based Superpixel Algorithms

SEAL [15] employs DNNs to acquire pixel affinity, inputting these affinities into ERS for superpixel generation. SEN [16] focuses on learning texture pattern similarity using a deep network, and then applies the learned features to SNIC for superpixel segmentation. Pan *et al.* [17] introduce a fast lattice superpixel generation, merging a deep network with soft K-means to generate superpixels possessing a lattice topology. Nevertheless, it is crucial to note that these methodologies incorporate non-differentiable operations and are not end-to-end trainable deep networks. SSN [11] is the first end-to-end trainable network architecture for superpixel segmentation. SSN is designed to learn pixel features which are then fed to a differentiable K-means clustering module. It comprises three scales, each equipped with two convolutional layers for computational processing. The final layer merges each scale's output and the initial input of the entire network, and outputs the required learned features.

LNS-Net [18] is an unsupervised CNN-based method dedicated to the non-iterative and lifelong acquisition of superpixels. However, the space transformation and seed distribution are image-level processing, thereby posing challenges in achieving ultra-low latency. Suzuki [19] employs CNN for the unsupervised generation of superpixels with regular information maximization. The entropy calculation of each superpixel poses challenges for FPGA implementation. SCN [12] is a fully-connected convolutional network that adopts an encoder-decoder structure, which simplifies the iterative clustering step of SSN by assigning each pixel into the 9-neighbor grid. AI-Net [13] achieves state-of-the-art performance by proposing an association implantation module, which provides consistent pixel-superpixel level context for the superpixel segmentation task. Over-SegNet [20] comprises an encoder and a decoder, designed for feature representation and pixel–superpixel association, respectively. The decoder incorporates a multi-scale convolutional structure with cross-large-scale connections to facilitate pixel–superpixel association in a coarse-to-fine feed-forward manner. Notably, the encoder-decoder structure demands a substantial amount of parameters and computational

resources, making its implementation as a hardwired type architecture challenging.

### 2.2 FPGA Implementation of Superpixel Algorithms

SS [21] is proposed for FPGA implementation of SEEDS. This method partitions the image into a lattice shape and employs an energy function for boundary updates. This approach achieves a throughput of 42.2 FPS. Akagic *et al.* [22] realize SLIC segmentation with a delay of 39.63 ms. However, the requirement for multiple transfers between the host and the device imposes limitations on the processing speed. Khamaneh *et al.* [23] introduce a memory-efficient architecture for SLIC, where the memory is designed to store the label of each pixel and RGB color information exclusively. The proposed architecture attains a frame rate of 24 FPS when applied to a camera with a resolution of 300×400 pixels. Mighani *et al.* [24] present a framework that substitutes the cluster-based search operation with a pixel-based process during the assignment step. The proposed architecture achieves a processing speed of 143 FPS for 300×400 resolution. Nonetheless, these works remain iterations of SLIC. Iterations necessitate frequent reading and writing of labels to memory, consequently resulting in long processing delays. FP-SLIC [25] adopts a strategy aimed at diminishing the number of iterations. Limiting the iterations to two times and implementing a fully pipelined FPGA architecture, the system achieves a processing delay of 3.86 ms for an image with a resolution of 481×321. Despite this reduction in delay, it falls short of meeting the requisites for ultra-low latency applications. Furthermore, the reduction of iteration times is accomplished at the expense of system robustness. [9] introduces FPGA-oriented algorithms for 1-ms SLIC and [10] extends this work and realizes system-level hardware implementation. 1-ms SLIC segregates iterations into the temporal domain. Through a single processing within each frame, the whole system attains a delay of 0.985 ms. But this work primarily focuses on the hand-crafted SLIC part. To combine deep network with SLIC to reach heightened performance and improved integration with other deep network tasks, further research is needed in the development of a 1-ms SSN-based superpixel segmentation system.

### 2.3 CNN FPGA Implementation

FPGAs are widely utilized for accelerating neural networks. Zhang *et al.* [26] introduce a double signed-multiplication correcting circuit to reduce the computational time of CNNs. WinoNN [27] introduces an efficient encoding format to minimize the online encoding overhead caused by activation sparsity. However, the reliance on external memory makes it challenging to improve the overall latency of the system. Some approaches focus on reducing memory access. Li *et al.* [28] use a kernel partition technique to reduce repeated access to input feature maps and kernels, achieving a processing time of 6.279 ms for AlexNet. Xuan *et al.* [29] propose a dataflow to process the depthwise separable convolution layer end-to-end, reducing memory accesses for intermediate feature maps. Yan *et al.* [30] propose multiple parallel strategies for different convolution types of MobileNet, with a processing time of 3.31 ms. A flexible and efficient FPGA accelerator [31] allows depthwise convolution to be executed directly after standard convolutions without external memory access, achieving a processing time of 4.5 ms for MobileNetV2. Nguyen *et al.* [32] employ mixed precision and mixed data flow, resulting in a latency of 9.15 ms, with off-chip access reduced to 0 Mb through the line buffer pipeline. However, intermediate results and parameters still need to be stored in BRAM inside the FPGA. Although these structures implement deep networks within several milliseconds, the constrained memory bandwidth of external and internal memory presents challenges in achieving ultra-low delay. For BNNs, LUTNet [33] and PolyLUT [34] take a different approach compared to methods relying on XNOR gates and additions. Instead, LUTNet encodes weights in its LUT masks, leading to greater logic density. PolyLUT goes even further by utilizing LUTs to represent polynomial evaluations. This approach allows for mapping sparse and quantized polynomial neural networks to netlists of LUTs, resulting in significant improvements in latency and area efficiency. In the pursuit of ultra-low delay CNN implementation, Zhang *et al.* [14] map CNNs directly on FPGA to avoid memory utilization. While it may limit the flexibility of models, it prioritizes ultra-low latency for specific applications.

## 3. Proposed Method

The comparisons between the structure of SSN and the proposed method are delineated in Fig. 1. In the SSN framework, 5-dimensional features encompassing both color and positional information serve as input to the deep network. These features traverse through three scale convolutional layers, ultimately reaching the final layer. Notably indicated by red lines, the final layer aggregates the outputs of all scales along with the initial input. The acquired learned features are subsequently utilized for the computation of pixel-superpixel associations. After obtaining all pixels' results, superpixel centers are updated. The updated center information is then employed for a subsequent iteration of pixel-superpixel association computation. This iterative process persists until a predefined number of iterations is reached. Not only iterations within differentiable SLIC contribute to a long delay, but the aggregated structure of the deep network renders the achievement of ultra-low latency unfeasible. The aggregated structure requires sustainable storage resources to accommodate all intermediate results from each scale and input. Even with sufficient resources, frequent memory access imposes constraints on the overall processing speed of the system. Most importantly, even if all these results are stored in registers and memory access is not a concern, the significant size of the final layer would still require extended processing times. This hinders fully pipelining and contributes to an overall increase in system latency.

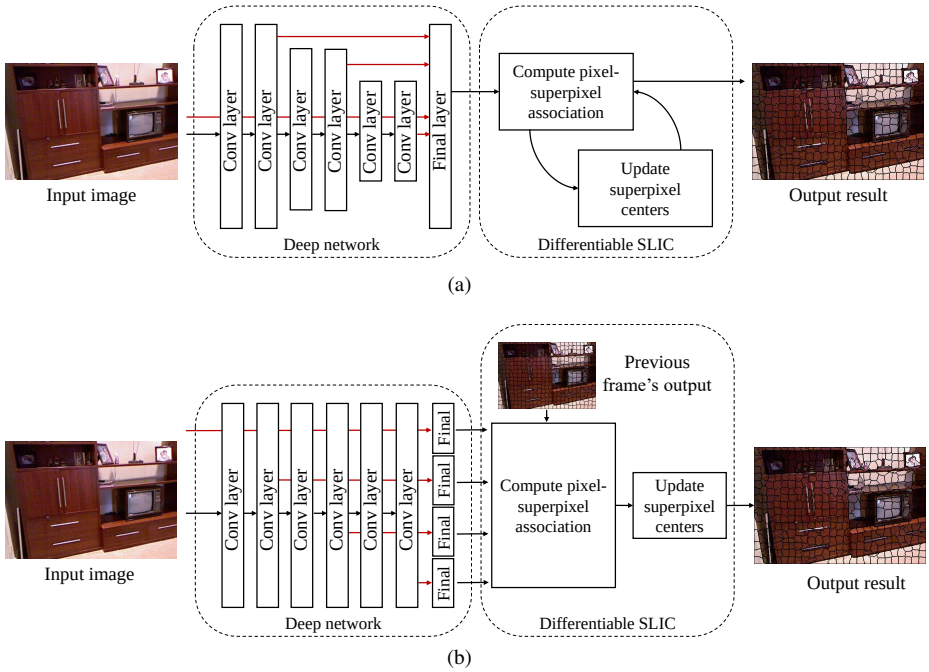



**Fig. 1** Comparisons between the structure of SSN and the proposed method. (a) SSN structure; (b) Proposed structure.

The structural representation of the proposed method is illustrated in Fig. 1(b). Inspired by 1-ms SLIC, the iterations within the SLIC part are separated into the temporal domain. The formulation of the proposed structure is articulated through the following equations.

$$\underset{n\times k}{F} = \mathcal{F}(I), \tag{1}$$

$$Q_{pi}^{cur} = e^{-\|F_p - S_i^{pre}\|^2}. \tag{2}$$

In the context of the entire image $\underset{n\times 5}{I}$, where $n$ denotes the number of pixels and each pixel encompasses 5-dimensional features, the learned features $\underset{n\times k}{F}$ from the deep network are derived. Here, $k$ represents the dimensions of the learned features, encompassing the original input's 5 dimensions. $\underset{m\times k}{S}$ embodies the features of $m$ superpixel centers. For the computation involving pixel $p$ and superpixel $i$, the pixel-superpixel association in the current frame, denoted as $Q_{pi}^{cur}$, is determined by Eq. 2. In this equation, $cur$ and $pre$ denote current frame and previous frame, respectively. The output from the previous frame, serving as superpixel center information, is utilized in computing the pixel-superpixel association for the current frame. Following a single processing iteration, the output result for the current frame is obtained.

To accommodate hardware resource constraints and align with the SSN architecture, the feature dimension is configured to 10. To capitalize on the information from each pixel, given the pixel stream characteristics of FPGA, a stride of 1 is employed between layers. Aggregated to pipelined structure is proposed in the deep network part. Layer-partitioned memory architecture is proposed for the FPGA implementation of differentiable SLIC with the streaming dataflow. Details are explained in the following subsections.

### 3.1 Aggregated to Pipelined Structure for FPGA Implementation

In the SSN framework, the final layer merges the initial input with the outputs from all three scales to derive learned features. Although adopting a hardwired implementation type eliminates the need to store weights, inputs, and outputs of each layer in memory, implementing the aggregated structure on an FPGA still falls short of achieving a 1-ms implementation. As depicted in Fig. 2(a), the final layer aggregates all intermediate results from each scale and input. The extended processing time poses challenges in realizing a pipelined structure. Moreover, to accommodate the simultaneous input of all data into the final layer, substantial memory

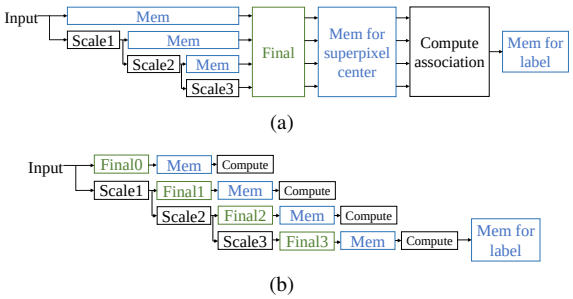



**Fig. 2** Conceptual difference. (a) Aggregated structure of SSN; (b) Proposed pipelined structure for FPGA implementation.

**Table 1** Proposed network structure.

| Layer Name | Input Source | Output Channels | Layer Name | Input Source | Output Channels |
|---|---|---|---|---|---|
| Final0 | Input | 2 | Conv3 | Conv2 | 16 |
| Conv0 | Input | 16 | Final2 | Conv3 | 1 |
| Conv1 | Conv0 | 16 | Pool1 | Conv3 | 16 |
| Final1 | Conv1 | 1 | Conv4 | Pool1 | 16 |
| Pool0 | Conv1 | 16 | Conv5 | Conv4 | 16 |
| Conv2 | Pool0 | 16 | Final3 | Conv5 | 1 |

resources become a must. This storage not only consumes significant resources but also introduces speed limitations due to frequent memory access, making a pipelined structure impractical. In conclusion, the aggregated structure presents challenges in realizing a pipelined structure, hindering the achievement of ultra-low delay.

For the proposed pipelined structure, the original SSN network structure undergoes a change. Specifically, the final layer is decomposed into several individual final layers, as illustrated in Fig. 2(b). Each scale, along with the input, is allocated an individual final layer, obviating the necessity to store intermediate results. This decomposition of the final layer mitigates pipeline blockage during the final layer processing, thereby enabling the realization of ultra-low latency. After processing each final layer, the respective dimensions of superpixel center information are read from memory for pixel-superpixel association computation. The entire structure operates in a fully pipelined manner. Once the association computation for the last final layer is completed, the label of each pixel is written into memory for the label map. The update of superpixel center information occurs after the computation of the entire frame. The original SSN final layer comprises 5 channels designed for 5-dimensional learned features. To enhance information extraction from the input and retain more input details, the final layer responsible for handling the input is assigned two channels, while the remaining three final layers each have one channel. The proposed pipelined network structure is elucidated in Table 1.

### 3.2 Layer-Partitioned Memory Architecture

Building upon the transition from an aggregated to a pipelined structure, the final layer is subdivided into several individual final layers. This enables the entire system to achieve full pipelining. To ensure that the system's delay from sensing to feedback remains within 1 ms, streaming processing is necessary with the pixel stream input. With the fully pipelined structure and streaming processing, each layer of the network simultaneously processes different pixel information. The deployment of a single memory unit to store all dimensions of superpixel center features for pixel-superpixel association computation often leads to frequent memory reading conflicts. The adoption of a sequential superpixel center reading strategy becomes a must under these circumstances. Nonetheless, this strategy imposes limitations on the processing speed of the entire system, presenting challenges in attaining a processing delay of 1 ms.

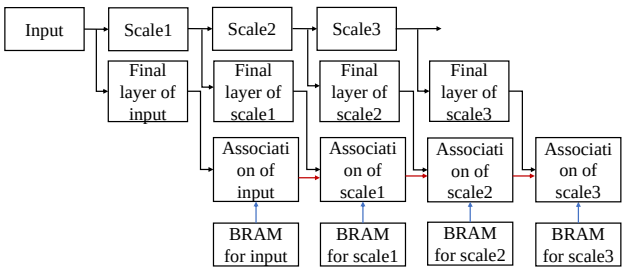



**Fig. 3** Layer-partitioned memory architecture.

To align with the pipelined structure, a layer-partitioned memory architecture is proposed. This architecture involves the division of memory dedicated to superpixel center features based on individual final layers. For instance, the first final layer is configured to generate 2-dimensional learned features from the input. Within the association computation module of this final layer, a dedicated memory unit is allocated to store the 2-dimensional learned features of the superpixel centers from the previous frame. Upon the arrival of the pixel stream at this final layer's association computation stage, the center information corresponding to this pixel is read from the allocated memory. Upon completion of the association map for the entire frame, the memory is utilized to update the 2-dimensional learned features of the superpixel centers for the current frame. The representation of the layer-partitioned memory architecture is depicted in Fig. 3.

Utilizing this memory architecture, pixel-superpixel associations are computed as the pixel stream traverses the structure. This approach effectively mitigates memory operation conflicts. Additionally, as indicated by the red arrows in Fig. 3, computation results are accumulated. The association result for a given pixel is derived once the pixel stream reaches the association computation stage of the last final layer. The entire proposed system is fully pipelined for the streaming dataflow.

### 3.3 Hardware Implementation

The proposed system is implemented on the hardware architecture illustrated in Fig. 4. The components include the BASLER acA2000-340 camera and the ZCU104 FPGA. For the specifications of the high frame rate camera, to attain the frame rate of 1000 FPS, each image has a maximum of 360 lines. Increasing the number of lines in the input image

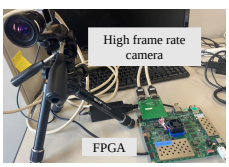



**Fig. 4** Components of the high frame rate and ultra-low delay system for realworld applications [10]. High frame rate camera is BASLER acA2000-340, FPGA is AMD Xilinx Zynq UltraScale+ MPSoC ZCU104.

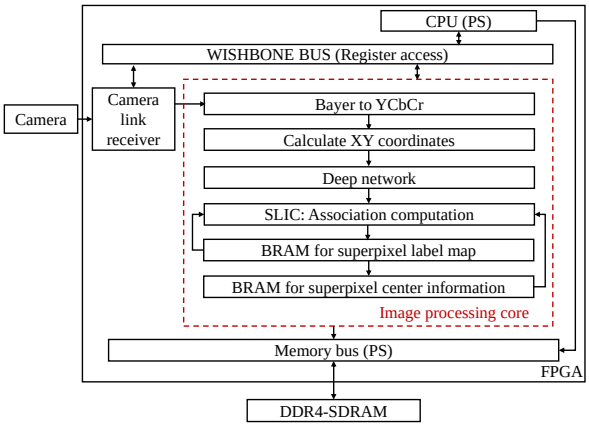



**Fig. 5** Hardware structure of the proposed streaming SSN superpixel segmentation system.

leads to a corresponding increase in transmission time for the camera, causing it to fail to meet the requirement of 1 ms. The hardware structure of the proposed streaming SSN superpixel segmentation system is depicted in Fig. 5. The processing system (PS) is implemented on one chip with programmable logic. The incoming from the camera undergoes a conversion process through the camera link receiver, resulting in a pixel stream operating at a frequency of 300 MHz. This processed pixel stream serves as the input for the image processing core within the FPGA. The output result of the image processing core is the label assigned to each pixel, which is transmitted to the PS for subsequent post-processing via the WISHBONE BUS. Furthermore, DDR4-SDRAM is available as external memory for potential post-analysis tasks. The superpixel segmentation system is implemented in the image processing core of the FPGA. The pixel stream is first transformed to YCbCr color space. While the original SSN framework utilizes the CIELAB color space, the RGB-to-CIELAB conversion involves computationally intensive operations such as divisions and exponentials. To preserve the perceptual lightness channel without computationally intensive mathematical operations, YCbCr color space is employed. Following the determination of positional coordinates for each pixel, the resulting 5-dimensional features serve as the input to the deep network.

To implement the deep network on the FPGA, floating-point operations are quantized to 8-bit integers utilizing QAT [35], a commonly employed and effective technique. In adapting the original network structure for hardware implementation, characterized as a hardwired type, the number of channels per layer is reduced from 64 to 16. Further parameter reduction is achieved through the adoption of depthwise separable convolution [36]. Upon completing the calculation of each final layer, the corresponding learned features are input into the SLIC module to compute with the surrounding nine superpixels. The outcomes are stored in a register for accumulation alongside results from other final layers. As the pixel stream reaches the last final layer, the label for each pixel is determined based on the smallest association among the surrounding superpixels. This label is then stored in the BRAM for the superpixel label map. After the completion of the entire frame's label assignment, the superpixel center information is updated and stored in BRAM.

The superpixel label map and superpixel center information from the previous frame serve as inputs for the association computation in the current frame. A label map is instrumental in determining the surrounding nine superpixel centers corresponding to a given pixel. BRAM for superpixel center information is divided into individual BRAMs based on layer-partitioned memory architecture. A ping-pong BRAM structure is implemented for both BRAMs, allowing simultaneous read access to the previous frame's results and write access to the current frame's results. Upon the completion of reading and writing operations for the current frame, the roles of these BRAMs are exchanged in preparation for the next frame.

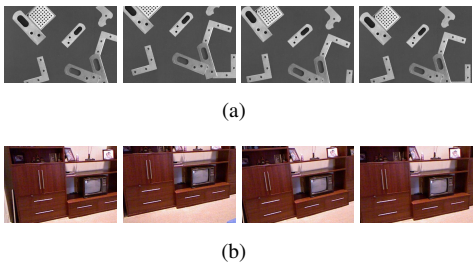
## 4. Experimental Results

### 4.1 Algorithm Evaluation

#### 4.1.1 Dataset and Evaluation Metrics

The experiments are conducted on both factory assembly line dataset and indoor scenes dataset to demonstrate the efficacy in the fields of factory automation and robotics. Consistent with the settings of 1-ms SLIC [10], component images from the Halcon example images [37] are employed to generate factory assembly line dataset. For simulating indoor scenes relevant to robotics applications, images from the NYUV2 dataset [38] are utilized. Both datasets feature videos with a resolution of 500×340 pixels. Each dataset has four different motion patterns including horizontal translation, vertical translation, rotation, and scale change. Examples of these datasets are visually presented in Fig. 6.

To objectively evaluate the segmentation results, three standard metrics are employed in this work. High boundary quality is essential for image segmentation [39], as various downstream applications directly benefit from more precise boundaries [40]. Therefore, as suggested by previous work [7], the most important property of a superpixel method is its ability to adhere to image boundaries. Boundary Recall



(a)

(b)

**Fig. 6** Examples of datasets [10]. (a) Factory assembly line dataset; (b) Indoor dataset. Horizontal and vertical translation datasets move at 1 pixel/frame; Rotation dataset rotates at 0.1 degree/frame; Scale dataset scales at 0.001 times/frame.

**Table 2** Evaluation results on indoor datasets.

| | Horizontal translation | | | Vertical translation | | | Rotation | | | Scale change | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | [10] | SSN | Proposed | [10] | SSN | Proposed | [10] | SSN | Proposed | [10] | SSN | Proposed |
| BR ↑ | 0.825 | 0.859 | 0.870 | 0.807 | 0.879 | 0.886 | 0.814 | 0.889 | 0.879 | 0.812 | 0.878 | 0.883 |
| UE ↓ | 0.057 | 0.044 | 0.046 | 0.057 | 0.041 | 0.042 | 0.063 | 0.049 | 0.049 | 0.062 | 0.049 | 0.047 |
| ASA ↑ | 0.943 | 0.956 | 0.954 | 0.943 | 0.959 | 0.958 | 0.937 | 0.951 | 0.951 | 0.938 | 0.951 | 0.953 |
| CO ↑ | 0.350 | 0.325 | 0.302 | 0.360 | 0.335 | 0.306 | 0.370 | 0.338 | 0.311 | 0.380 | 0.336 | 0.318 |
| LC ↑ | 0.858 | 0.830 | 0.871 | 0.839 | 0.794 | 0.841 | 0.915 | 0.944 | 0.955 | 0.813 | 0.798 | 0.824 |

**Table 3** Evaluation results on factory assembly line datasets.

| | Horizontal translation | | | Vertical translation | | | Rotation | | | Scale change | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | [10] | SSN | Proposed | [10] | SSN | Proposed | [10] | SSN | Proposed | [10] | SSN | Proposed |
| BR ↑ | 0.990 | 0.999 | 0.991 | 0.979 | 0.998 | 0.994 | 0.994 | 0.998 | 0.994 | 0.991 | 0.998 | 0.994 |
| UE ↓ | 0.018 | 0.013 | 0.015 | 0.020 | 0.013 | 0.014 | 0.024 | 0.019 | 0.021 | 0.024 | 0.019 | 0.021 |
| ASA ↑ | 0.982 | 0.987 | 0.985 | 0.980 | 0.987 | 0.986 | 0.976 | 0.981 | 0.979 | 0.976 | 0.981 | 0.979 |
| CO ↑ | 0.445 | 0.457 | 0.475 | 0.457 | 0.459 | 0.481 | 0.465 | 0.456 | 0.478 | 0.468 | 0.457 | 0.482 |
| LC ↑ | 0.877 | 0.847 | 0.908 | 0.880 | 0.863 | 0.909 | 0.943 | 0.958 | 0.974 | 0.847 | 0.833 | 0.870 |

(BR) and Under-segmentation Error (UE) are standard measures for evaluating boundary adherence [41]. Achievable Segmentation Accuracy (ASA) is applied to estimate the upper bound on the achievable segmentation accuracy when utilizing superpixel as a pre-processing step. Additionally, Compactness (CO) is employed to evaluate the visual quality of the segmentation algorithms. Label consistency (LC), as defined in [42], is employed to evaluate the stability of superpixels across consecutive frames in a video sequence. LC shows how effectively superpixels track parts of objects over time, playing a pivotal role in subsequent tasks such as classification or tracking. When label consistency is low, it may lead to unstable or flickering classification outcomes, significantly impacting robotic tasks in practical scenarios [43].

#### 4.1.2 Evaluation Results and Analysis

Comparisons among 1-ms SLIC, SSN, and the proposed method are detailed in Table 2 and Table 3. In these experiments, the expected number of superpixels is set to 400. The proposed system demonstrates comparable values of BR, UE, and ASA for both datasets comparing with SSN. However, due to the aggregated structure of SSN, it is impossible to be implemented on FPGA to achieve ultra-low delay. Despite the decrease in CO observed in indoor dataset, the proposed system exhibits an average increase of more than 3.74 % in LC in comparison to SSN. As the results are utilized for feedback in factory automation and robotics scenarios, CO, which reflects visual quality, only serves as a supplementary evaluation metric. With the integration of a deep network, the proposed method improves BR of the indoor dataset by an average of 7.89 % compared to the 1-ms SLIC method. This demonstrates the proposed method's ability to adhere to boundaries in complex scenes. The less pronounced improvement in the FA dataset is due to its already clear boundaries against a simple background, resulting in a BR of average 0.989 for the 1-ms SLIC method. The proposed method achieves an average improvement in UE of 24.19 % for the indoor dataset and 17.92 % for the FA dataset. These improvements in BR and UE highlight the significant enhancement in boundary adherence achieved by the proposed method compared to the previous 1-ms SLIC method. Notably, the proposed method even outperforms in LC compared to 1-ms SLIC.

To highlight the efficacy of the proposed methods, thorough comparisons are conducted with other well-known algorithms. Quantitative experiments are performed, utilizing the horizontal translation of the indoor dataset as an illustrative example. In addition to SSN, the comparative analysis involves various hand-crafted algorithms, including SLIC [7], SNIC [44], SEEDS [45], LSC [8], ERS [46], ETPS [47], and 1-ms SLIC [10]. The red line in Fig. 7 signifies the performance of the proposed method. Among all algorithms, ERS exhibits superior performance across the metrics of BR, UE, and ASA. Nonetheless, owing to its reliance on global processing and complex calculations, the implementation of ERS within an ultra-low delay system poses inherent challenges. Excluding ERS, the proposed method shows superior performance in BR, UE, and ASA compared with other algorithms and is comparable to SSN. In terms of CO, ETPS demonstrates the best performance. However, it does not perform well in the other evaluation metrics. Given that the network places a stronger emphasis on boundary adherence, CO yields only moderate results. Nevertheless, in terms of LC, the proposed method achieves the highest performance.

Comparisons with other deep network-based algorithms are additionally carried out on indoor datasets. The experiments involve the utilization of pre-trained modules from SEAL, SCN, and AINet. To uphold fairness in the evaluations, the expected number of superpixels of the proposed system is standardized to 750. The evaluation results are presented in Table 4. While SEAL exhibits excellent performance in BR, it demonstrates a significant decrease compared to other works in the other four evaluation metrics. Although the proposed method is primarily designed for FPGA implementation to achieve ultra-low delay, it shows comparable performance with other deep learning methods

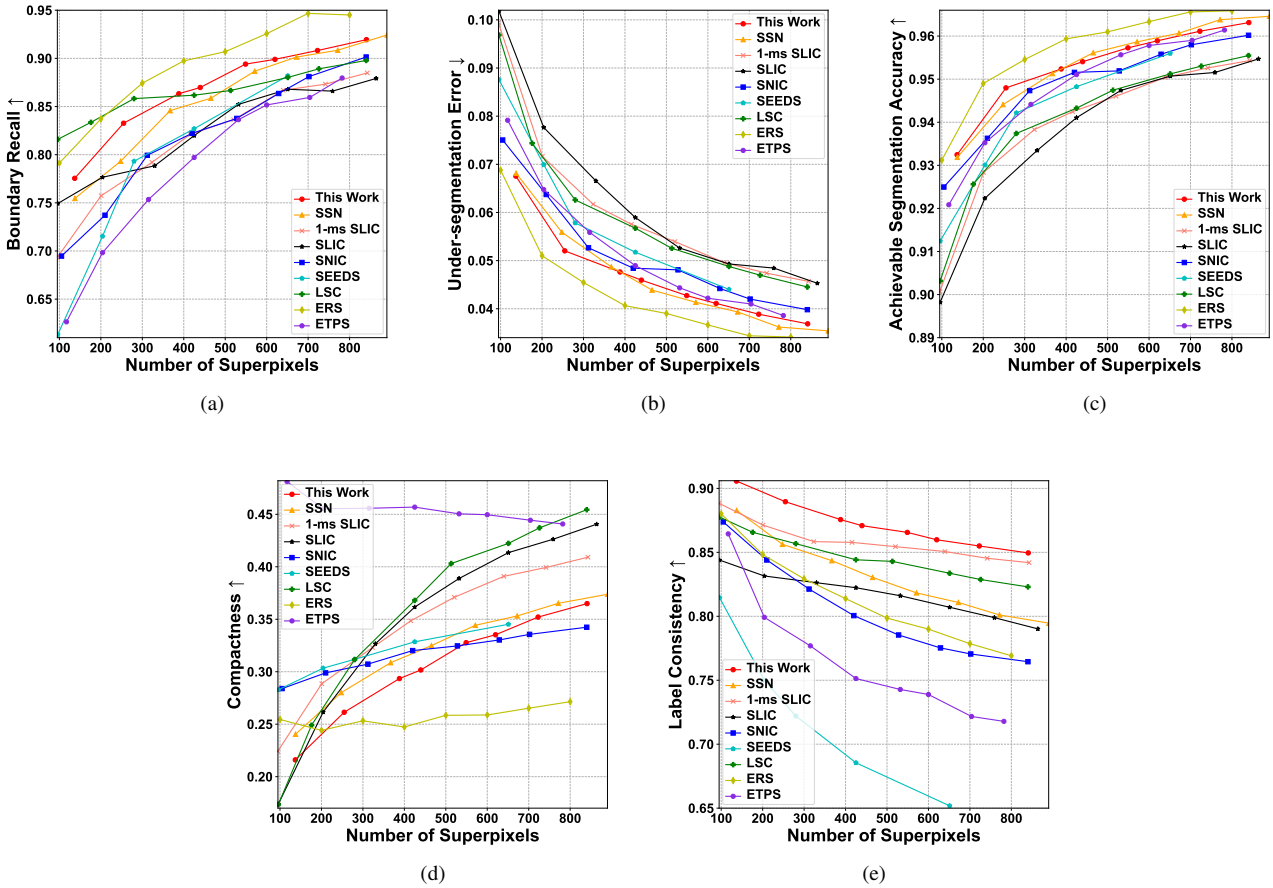



**Fig. 7** Quantitative evaluation results. (a) Boundary recall; (b) Under-segmentation error; (c) Achievable segmentation accuracy; (d) Compactness; (e) Label consistency.

**Table 4** Comparisons with deep learning algorithms in indoor dataset.

| | Horizontal translation | | | | Vertical translation | | | | Rotation | | | | Scale change | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SEAL | SCN | AINet | This work | SEAL | SCN | AINet | This work | SEAL | SCN | AINet | This work | SEAL | SCN | AINet | This work |
| BR ↑ | 0.948 | 0.892 | 0.892 | 0.908 | 0.947 | 0.927 | 0.903 | 0.918 | 0.952 | 0.919 | 0.903 | 0.922 | 0.954 | 0.916 | 0.908 | 0.918 |
| UE ↓ | 0.085 | 0.035 | 0.036 | 0.039 | 0.085 | 0.031 | 0.034 | 0.037 | 0.093 | 0.040 | 0.043 | 0.042 | 0.092 | 0.041 | 0.043 | 0.043 |
| ASA ↑ | 0.915 | 0.965 | 0.964 | 0.961 | 0.915 | 0.969 | 0.966 | 0.963 | 0.907 | 0.960 | 0.957 | 0.958 | 0.908 | 0.959 | 0.957 | 0.957 |
| CO ↑ | 0.121 | 0.369 | 0.350 | 0.352 | 0.121 | 0.373 | 0.353 | 0.357 | 0.122 | 0.369 | 0.348 | 0.361 | 0.121 | 0.369 | 0.349 | 0.360 |
| LC ↑ | 0.807 | 0.843 | 0.823 | 0.855 | 0.769 | 0.806 | 0.786 | 0.824 | 0.955 | 0.948 | 0.933 | 0.953 | 0.633 | 0.781 | 0.762 | 0.792 |

in BR. Despite an inferior performance in UE, ASA, and CO, the proposed method excels LC, a crucial metric highlighting the stability of superpixels.

### 4.1.3 Ablation Study

In the proposed aggregated to pipelined structure for FPGA implementation, two channels are allocated to the final layer to capture more information from the input. Ablation studies are performed on alternative decomposed network structures. f0 denotes assigning two channels to final0. While f1, f2, and f3 correspond to assigning two channels to final1, final2 and final3, respectively. The experiments are conducted using the indoor dataset as an example, and the results are presented in Table 5. Due to the enhanced extraction of input information, the adopted structure exhibits better boundary adherence and segmentation accuracy.

To further demonstrate the robustness and applicability of the proposed method, we examined the relationship between image size and superpixel segmentation performance using horizontal translation of the indoor dataset as an example. The evaluation results are presented in Table 6. Larger image sizes result in higher BR for the same desired number of superpixels, while increasing image sizes decreases segmentation performance. Conversely, smaller image resolutions yield more compact superpixels. Regarding label

**Table 5** Ablation studies of different decomposed network structures.

| | Horizontal translation | | | | Vertical translation | | | | Rotation | | | | Scale change | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | f0 | f1 | f2 | f3 | f0 | f1 | f2 | f3 | f0 | f1 | f2 | f3 | f0 | f1 | f2 | f3 |
| BR ↑ | 0.870 | 0.861 | 0.861 | 0.861 | 0.886 | 0.885 | 0.885 | 0.885 | 0.879 | 0.879 | 0.879 | 0.879 | 0.883 | 0.875 | 0.875 | 0.875 |
| UE ↓ | 0.046 | 0.046 | 0.046 | 0.046 | 0.042 | 0.043 | 0.043 | 0.043 | 0.049 | 0.048 | 0.048 | 0.048 | 0.047 | 0.049 | 0.049 | 0.049 |
| ASA ↑ | 0.954 | 0.954 | 0.954 | 0.954 | 0.958 | 0.957 | 0.957 | 0.957 | 0.951 | 0.952 | 0.952 | 0.952 | 0.953 | 0.951 | 0.951 | 0.951 |
| CO ↑ | 0.302 | 0.309 | 0.309 | 0.309 | 0.306 | 0.312 | 0.312 | 0.312 | 0.311 | 0.315 | 0.315 | 0.315 | 0.318 | 0.356 | 0.326 | 0.326 |
| LC ↑ | 0.871 | 0.873 | 0.873 | 0.873 | 0.841 | 0.845 | 0.845 | 0.845 | 0.955 | 0.955 | 0.955 | 0.955 | 0.824 | 0.825 | 0.825 | 0.825 |

**Table 6** Relationship between different image resolutions and superpixel segmentation performance.

| Number of lines | BR ↑ | UE ↓ | ASA ↑ | CO ↑ | LC ↑ |
|---|---|---|---|---|---|
| 200 | 0.716 | 0.029 | 0.971 | 0.429 | 0.880 |
| 300 | 0.767 | 0.039 | 0.961 | 0.358 | 0.874 |
| 400 | 0.880 | 0.043 | 0.957 | 0.304 | 0.877 |


**Fig. 8** Examples of cluttered datasets.

consistency, all demonstrate similar performance levels.

#### 4.1.4 Experiments on cluttered dataset

To broaden the applicability of the proposed method, several cluttered images from NYUV2 are adopted to generate testing datasets. The generation strategy for these datasets mirrors that of the indoor dataset. Fig. 8 showcases examples of the cluttered datasets, featuring office desk, bookshelf, bedroom, and storage room scenes, respectively.

The evaluation results for the cluttered datasets are presented in Table 7. Across all datasets, the boundary recall exceeds 0.83, and achievable segmentation accuracy surpasses 0.89. This demonstrates the applicability of the proposed methods to a wide range of applications.

### 4.2 Hardware Evaluation

Processing speed and the utilization of hardware resources are pivotal concerns, given that the ultimate objective of this work is the development of a 1-ms superpixel segmentation system. The synthesis and implementation are performed on the ZCU104 FPGA board utilizing Vivado 2021.2. BRAM is configured to support a capacity of up to 512 superpixels. Even with the reduced 16-channel configuration and 8-bit integer calculations, the original aggregated structure of SSN still requires 65.3 Mb of BRAM resources. Implementing it without external memory and achieving ultra-low latency are impossible. Therefore, comparisons were conducted with 1-ms SLIC, and the hardware evaluation results are presented in

**Table 7** Evaluation results on cluttered datasets.

| Dataset | Motion | BR ↑ | UE ↓ | ASA ↑ | CO ↑ | LC ↑ |
|---|---|---|---|---|---|---|
| Office Desk | Horizontal translation | 0.832 | 0.088 | 0.912 | 0.321 | 0.853 |
| | Vertical translation | 0.843 | 0.085 | 0.915 | 0.315 | 0.810 |
| | Rotation | 0.854 | 0.093 | 0.907 | 0.319 | 0.939 |
| | Scale change | 0.842 | 0.094 | 0.906 | 0.323 | 0.788 |
| Bookshelf | Horizontal translation | 0.902 | 0.076 | 0.924 | 0.267 | 0.747 |
| | Vertical translation | 0.926 | 0.070 | 0.930 | 0.271 | 0.812 |
| | Rotation | 0.921 | 0.085 | 0.915 | 0.265 | 0.914 |
| | Scale change | 0.923 | 0.088 | 0.912 | 0.265 | 0.731 |
| Bedroom | Horizontal translation | 0.845 | 0.079 | 0.921 | 0.320 | 0.841 |
| | Vertical translation | 0.869 | 0.077 | 0.923 | 0.321 | 0.832 |
| | Rotation | 0.863 | 0.086 | 0.914 | 0.319 | 0.949 |
| | Scale change | 0.863 | 0.083 | 0.917 | 0.319 | 0.792 |
| Storage Room | Horizontal translation | 0.871 | 0.096 | 0.904 | 0.282 | 0.841 |
| | Vertical translation | 0.870 | 0.093 | 0.907 | 0.282 | 0.788 |
| | Rotation | 0.878 | 0.108 | 0.892 | 0.277 | 0.941 |
| | Scale change | 0.886 | 0.102 | 0.898 | 0.278 | 0.768 |

**Table 8** Hardware performance and resource utilization of the proposed system and 1-ms SLIC.

| Item | | [10] | Proposed |
|---|---|---|---|
| Resource utilization | #LUT | 177618 (77.9%) | 163330 (70.89%) |
| | #LUTRAM | 3281 (3.22%) | 52745 (51.83%) |
| | #FF | 227166 (49.30%) | 104665 (22.71%) |
| | #BRAM | 188 (60.26%) | 311.5 (99.84%) |
| | #DSP | 210 (12.15%) | 26 (1.5%) |
| Performance | Frequency | 300 MHz | 300 MHz |
| | Delay per frame | 0.981 ms | 0.947 ms |

Table 8. For the 1-ms SLIC, operating at a clock frequency of 100 MHz, the delay is 0.985 ms per frame. However, when the clock frequency is increased to 300 MHz, only the calculation time is reduced, as the transmission time for the camera remains constant. Consequently, under the 300 MHz scenario, the processing time decreases to 0.981 ms per frame.

As the entire deep network is directly mapped onto the FPGA in a hardwired type for a 1-ms delay, the utilization

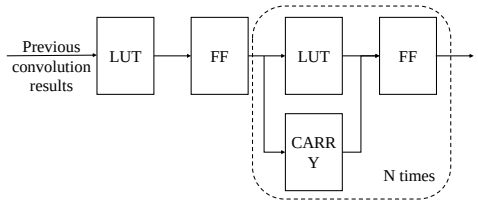



**Fig. 9** Schematic diagram of the proposed system.

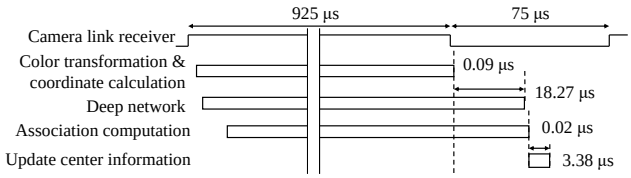


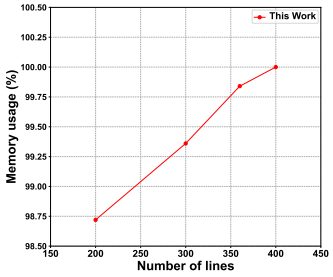

**Fig. 10** Detailed timing flow of the proposed system.

of LUT exceeds 70 %. The simplified schematic diagram of the proposed system is depicted in Fig. 9. The parameter $N$ in this figure depends on the number of addition operations, which correlates with the channel number. Multiplications are mapped to LUTs, while additions are mapped to both LUTs and CARRYs. FFs are utilized to ensure the synchronization of each calculation. The hardwired implementation fully leverages LUTs resources to execute the calculations of the neural network.

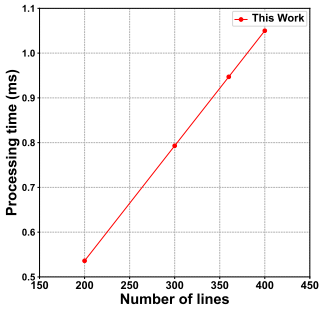
Because of the streaming structure of the proposed superpixel segmentation system, the computational resources only depend on the dimensions of the learned features in the deep network part. BRAM is employed for storing the label map and superpixel center information, with a ping-pong BRAM structure being adopted. Consequently, the utilization of BRAM has nearly reached its upper limit. Notably, for the proposed method, the superpixel center necessitates 10-dimensional features, whereas 1-ms SLIC only requires positional information. Consequently, the BRAM utilization for the proposed method surpasses that of 1-ms SLIC. The proposed system, serving as the pre-processing stage, has already utilized the majority of the hardware resources in pursuit of achieving a 1 ms delay. To integrate with subsequent processes, a FPGA with greater resources would be necessary. Alternatively, deploying multiple FPGAs is another available option, facilitated by the reduced data transmission resulting from superpixel segmentation.

Operated at a frequency of 300 MHz, the processing delay for the proposed system is 0.947 ms per frame. The detailed timing flow is illustrated in Fig. 10. Predominantly, the computational workload is centered around the deep network calculations. For the deep network part, it takes 18.27 µs from the first layer to the last final layer. Subsequently, the update of superpixel center information commences after the completion of pixel-superpixel association computation, requiring 3.38 µs.

Fig. 11 shows the impact of image sizes on hardware performance. Because the industrial camera's specification




(a)




(b)

**Fig. 11** Impact of image sizes on hardware performance. (a) Memory usage; (b) Processing speed.

dictates a fixed transmission time for each line, the number of lines varies in the experiments. Increasing the image size primarily impacts the memory allocated for storing the label map. Since the majority of memory usage is occupied by resources for storing superpixel center information, the impact of increasing image size on overall memory usage is not significant. However, when the number of lines reaches 400, the memory of the FPGA board is fully utilized, reaching 100 %. Regarding processing speed, as the number of lines in the image increases, the processing time also increases. When the number of lines reaches 400, the processing time exceeds 1.050 milliseconds, failing to meet the requirement for ultra-low delay. Increasing the number of lines in the input image results in a proportional increase in processing time for the proposed algorithms, further deviating from the 1 ms requirement.

To integrate with other DNN tasks, the weights can be changed using the task-specific reconstruction loss function. For instance, when combining with semantic segmentation networks, semantic labels can be included as pixel properties in the reconstruction loss to prompt SSN to learn superpixels aligned with semantic segments. Only the weights need to be adjusted in the hardwired type implementation for inference for different applications. In order to improve real-world applicability, the computational efficiency test on the GPU is also conducted. Utilizing the RTX 2080, the original SSN has an inference time of 0.254 seconds per frame, whereas the proposed method achieves an inference time of 0.099 seconds per frame. Despite being primarily tailored for

FPGA implementation, the proposed method proves to be effective on the GPU platform as well.

## 5. Conclusion

To realize a 1-ms SSN superpixel segmentation system, both algorithmic and hardware implementation have been presented in this paper. For the FPGA implementation of the deep network within SSN, an aggregated to pipelined structure is proposed. This structure involves the decomposition of the aggregation layer into multiple individual layers, thereby obviating the need for intermediate result storage. Sustainable memory resource requirements are eliminated, and the entire system is fully pipelined with pixel stream. Additionally, memory for storing superpixel center information is partitioned based on layers. The pixel outcomes are acquired as the pixel stream traverses through the system. Experimental results demonstrate that the proposed system achieves a real-time processing speed of 0.947 ms per frame. Moreover, its performance remains comparable to SSN and surpasses other well-known algorithms. In terms of LC, the proposed system reaches state-of-the-art performance. For future research, the current algorithm obtains commendable performance in boundary adherence and segmentation accuracy. However, the visual quality of the proposed algorithm is identified as an area for potential enhancement. Further research can prioritize the introduction of a loss function to improve CO results. While deploying multiple FPGAs is an option for subsequent processing, as the proposed method aids in reducing communication burdens, optimizing BRAM utilization further is necessary to enhance efficiency and resource consumption for subsequent processing tasks.

## Acknowledgments

This work was supported by KAKENHI (21K11816).

### References

[1] J.S. Rentmeister, M.H. Kiani, K. Pister, and J.T. Stauth, "A 120–330v, sub-$\mu$a, 4-channel driver for microrobotic actuators with wireless-optical power delivery and over 99% current efficiency," IEEE Symp. VLSI Circuits, pp.1–2, IEEE, 2020.

[2] S. Du, Y. Li, and T. Ikenaga, "Temporally forward nonlinear scale space for high frame rate and ultra-low delay a-kaze matching system," IEICE Trans. Inf. & Syst., vol.103, no.6, pp.1226–1235, 2020.

[3] T. Hu, R. Fuchikami, and T. Ikenaga, "High temporal resolution-based temporal iterative tracking for high framerate and ultra-low delay dynamic tracking system," IEICE Trans. Inf. & Syst., vol.105, no.5, pp.1064–1074, 2022.

[4] S. Du, Z. Dong, Y. Li, and T. Ikenaga, "Straight-line detection within 1 millisecond per frame for ultra-high-speed industrial automation," IEEE Trans. Ind. Informat., vol.19, no.4, pp.5965–5975, 2023.

[5] C. Zhao, W. Zhu, and S. Feng, "Superpixel guided deformable convolution network for hyperspectral image classification," IEEE Trans. Image Process., vol.31, pp.3838–3851, 2022.

[6] Z. Wu, H. Zhu, L. He, Q. Zhao, J. Shi, and W. Wu, "Real-time stereo matching with high accuracy via spatial attention-guided upsampling," Appl. Intell., pp.1–22, 2023.

[7] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," IEEE Trans. Pattern Anal. Mach. Intell., vol.34, no.11, pp.2274–2282, 2012.

[8] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.1356–1363, 2015.

[9] Y. Li, T. Hu, R. Fuchikami, and T. Ikenaga, "Grid sample based temporal iteration and compactness-coefficient distance for high frame and ultra-low delay slic segmentation system," Int. Conf. Mach. Vis. Appl. (MVA), 2023.

[10] Y. Li, T. Hu, R. Fuchikami, and T. Ikenaga, "Grid sample based temporal iteration for fully pipelined 1-ms slic superpixel segmentation system," IEICE Trans. Inf. & Syst., vol.107, no.4, pp.515–524, 2024.

[11] V. Jampani, D. Sun, M.Y. Liu, M.H. Yang, and J. Kautz, "Superpixel sampling networks," Proc. Eur. Conf. Comput. Vis. (ECCV), pp.352–368, 2018.

[12] F. Yang, Q. Sun, H. Jin, and Z. Zhou, "Superpixel segmentation with fully convolutional networks," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.13964–13973, 2020.

[13] Y. Wang, Y. Wei, X. Qian, L. Zhu, and Y. Yang, "Ainet: Association implantation for superpixel segmentation," Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp.7078–7087, 2021.

[14] P. Zhang, T. Hu, D. Luo, S. Du, and T. Ikenaga, "Highly-parallel hardwired deep convolutional neural network for 1-ms dual-hand tracking," IEEE Trans. Circuits Syst. Video Technol., vol.32, no.12, pp.8192–8203, 2022.

[15] W.C. Tu, M.Y. Liu, V. Jampani, D. Sun, S.Y. Chien, M.H. Yang, and J. Kautz, "Learning superpixels with segmentation-aware affinity loss," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.568–576, 2018.

[16] U. Gaur and B. Manjunath, "Superpixel embedding network," IEEE Trans. Image Process., vol.29, pp.3199–3212, 2020.

[17] X. Pan, Y. Zhou, Y. Zhang, and C. Zhang, "Fast generation of superpixels with lattice topology," IEEE Trans. Image Process., vol.31, pp.4828–4841, 2022.

[18] L. Zhu, Q. She, B. Zhang, Y. Lu, Z. Lu, D. Li, and J. Hu, "Learning the superpixel in a non-iterative and lifelong manner," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.1225–1234, 2021.

[19] T. Suzuki, "Superpixel segmentation via convolutional neural networks with regularized information maximization," IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP), pp.2573–2577, IEEE, 2020.

[20] P. Li and W. Ma, "Oversegnet: A convolutional encoder–decoder network for image over-segmentation," Comput. Electr. Eng., vol.107, p.108610, 2023.

[21] M. MIYAMA, "Fpga accelerator for super-pixel segmentation featuring clear detail and short boundary," IIEEJ Trans. Image Electron. Visual Comput., vol.5, no.2, pp.83–91, 2017.

[22] A. Akagic, E. Buza, R. Turcinhodzic, H. Haseljic, N. Hiroyuki, and H. Amano, "Superpixel accelerator for computer vision applications on arria 10 soc," IEEE Int. Symp. Design Diagnostics Electron. Circuits & Syst. (DDECS), pp.55–60, IEEE, 2018.

[23] P.A. Khamaneh, A. Khakpour, M. Shoaran, and G. Karimian, "Real–time memory efficient slic accelerator for low–power applications," Multimed. Tools. Appl., vol.81, no.22, pp.32449–32467, 2022.

[24] M. Mighani and A. Khakpour, "Fmslic: Fast memory-efficient structure for implementation of slic on fpga," Circuits, Syst. Signal Process., pp.1–14, 2023.

[25] A. Ghaderi, C. Ahlberg, M. Östgren, F. Ekstrand, and M. Ekström, "Fp-slic: A fully-pipelined fpga implementation of superpixel image segmentation," Euromicro Conf. Digit. Syst. Des. (DSD), pp.109–117, IEEE, 2022.

[26] J. Zhang, L. Cheng, C. Li, Y. Li, G. He, N. Xu, and Y. Lian, "A low-latency fpga implementation for real-time object detection," Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), pp.1–5, IEEE, 2021.

[27] X. Wang, C. Wang, J. Cao, L. Gong, and X. Zhou, "Winonn: Optimizing fpga-based convolutional neural network accelerators using

sparse winograd algorithm," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.39, no.11, pp.4290–4302, 2020.
[28] J. Li, K.F. Un, W.H. Yu, P.I. Mak, and R.P. Martins, "An fpga-based energy-efficient reconfigurable convolutional neural network accelerator for object recognition applications," IEEE Trans. Circuits Syst., II, Exp. Briefs, vol.68, no.9, pp.3143–3147, 2021.
[29] L. Xuan, K.F. Un, C.S. Lam, and R.P. Martins, "An fpga-based energy-efficient reconfigurable depthwise separable convolution accelerator for image recognition," IEEE Trans. Circuits Syst., II, Exp. Briefs, vol.69, no.10, pp.4003–4007, 2022.
[30] S. Yan, Z. Liu, Y. Wang, C. Zeng, Q. Liu, B. Cheng, and R.C. Cheung, "An fpga-based mobilenet accelerator considering network structure characteristics," Proc. Int. Conf. Field Program. Log. Appl. (FPL), pp.17–23, IEEE, 2021.
[31] X. Wu, Y. Ma, M. Wang, and Z. Wang, "A flexible and efficient fpga accelerator for various large-scale and lightweight cnns," IEEE Trans. Circuits Syst. I, Reg. Papers, vol.69, no.3, pp.1185–1198, 2021.
[32] D.T. Nguyen, H. Kim, and H.J. Lee, "Layer-specific optimization for mixed data flow with mixed precision in fpga design for cnn-based object detectors," IEEE Trans. Circuits Syst. Video Technol., vol.31, no.6, pp.2450–2464, 2020.
[33] E. Wang, J.J. Davis, P.Y. Cheung, and G.A. Constantinides, "Lutnet: Learning fpga configurations for highly efficient neural network inference," IEEE Trans. Comput., vol.69, no.12, pp.1795–1808, 2020.
[34] M. Andronic and G.A. Constantinides, "Polylut: learning piecewise polynomials for ultra-low latency fpga lut-based inference," Proc. Int. Conf. Field Programmable Technol. (ICFPT), pp.60–68, IEEE, 2023.
[35] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.2704–2713, 2018.
[36] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.4510–4520, 2018.
[37] Halcon, Available online: `https://linx.jp/product/mvtec/halcon/`.
[38] S. Song, S.P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.567–576, 2015.
[39] B. Cheng, R. Girshick, P. Dollár, A.C. Berg, and A. Kirillov, "Boundary iou: Improving object-centric image segmentation evaluation," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.15334–15342, 2021.
[40] D. Marmanis, K. Schindler, J.D. Wegner, S. Galliani, M. Datcu, and U. Stilla, "Classification with an edge: Improving semantic image segmentation with boundary detection," ISPRS J. Photogramm. Remote Sens., vol.135, pp.158–172, 2018.
[41] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation-supervoxels for point clouds," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.2027–2034, 2013.
[42] J. Chang, D. Wei, and J.W. Fisher, "A video representation using temporal superpixels," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.2051–2058, 2013.
[43] O. Miksik, D. Munoz, J.A. Bagnell, and M. Hebert, "Efficient temporal consistency for streaming video scene analysis," Int. Conf. Robot. Autom. (ICRA), pp.133–139, IEEE, 2013.
[44] R. Achanta and S. Susstrunk, "Superpixels and polygons using simple non-iterative clustering," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.4651–4660, 2017.
[45] M. Van den Bergh, X. Boix, G. Roig, and L. Van Gool, "Seeds: Superpixels extracted via energy-driven sampling," Int. J. Comput. Vision, vol.111, pp.298–314, 2015.
[46] M.Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.2097–2104, IEEE, 2011.
[47] J. Yao, M. Boben, S. Fidler, and R. Urtasun, "Real-time coarse-to-fine topologically preserving segmentation," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.2947–2955, 2015.

**Yuan Li** received the B.E. degree in information engineering from Southeast University, Nanjing, China, in 2018, and the M.E. degree in engineering of information, production and systems, in 2019, from Waseda University, Kitakyushu, Japan, where she is currently working towards the Ph.D. degree. Her research focuses on hardware implementation of computer vision algorithms.

**Tingting Hu** received her B.E. degree in School of Automation from the Beijing Institute of Technology, China, in 2016, and her M.E. and Ph.D. degrees from the Graduate School of Information, Production, and Systems of Waseda University, Japan, in 2017 and 2023, respectively. She joined Panasonic Corporation, Japan, in 2018. Her current research interests are high frame rate and ultra-low delay vision systems.

**Ryuji Fuchikami** received his B.Sc. degree in Kyushu Institute of Technology, Japan, in 1999. He joined Panasonic Corporation (Formerly Matsushita Electric Industrial Co., Ltd.), Japan, in 2000. He had been undertaking research on the LSI design of video codec. His current research interests are high-speed visual systems.

**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 integrated system 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. VVC, SCC), video filter (e.g. super resolution, high-dynamic range imaging), and video recognition (e.g. sports analysis, ultra-high speed and ultra-low delay vision system). 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).