Development of Haze Removing Hardware Using High-Level Synthesis

Daiki Shirai, Akira Yamawaki
[1,2]*Department of Engineering, Kyushu Institute of Technology, 1-1, Sensui-cho, Tobata-ku Kitakyushu-shi, Fukuoka, Japan*

E-mail: shirai.daiki885@mail.kyutech.jp, yama@ecs.kyutech.jp

Abstract

We have developed a hardware for low-power and high-speed haze removing to sharpen hazy images in battery-powered embedded image processing systems such as drones. For development, we used HLS (High-Level Synthesis), which automatically converts software into hardware. In this paper, we evaluate the performance and power efficiency of the hardware obtained by high-level synthesis based on the haze removing software developed considering the hardware specifications and show its effectiveness.

Keywords: Haze Removing, High-Level Synthesis, HLS, FPGA

1. Introduction

In recent years, the research and development of embedded devices using image recognition has been very active, and the market scale is rapidly expanding.

Among them, the use of aerial images taken by cameras mounted on drones and other UAV equipment is attracting a lot of attention, and research and projects related to it are being actively conducted. In Japan, where infrastructure facilities are aging rapidly and the workforce is shrinking, more and more facilities are being inspected using aerial images from drones.

In such cases, the aerial images are blurred by weather and shooting conditions, making it impossible to perform accurate inspections. In order to solve this problem, we decided to develop an image processing module that can be installed in the drone to remove the haze generated in the captured images in real time.

The module to be developed in this paper is a hardware that executes the haze removing based on the darkest channel priority method,[1, 2] and high-level synthesis is used for its development. In addition, to ensure that the hardware to be developed has high performance, we will also develop software programs that describe the processing contents and modify the algorithms in consideration of high-level synthesis.

Finally, the effectiveness of the developed hardware is demonstrated by comparing the hardware obtained by high-level synthesis with pure software following the original algorithm and software considering hardware.

2. Haze Removing

2.1. *Dark channel priority method*

The haze removing algorithm in this paper is based on the dark channel priority method. Normally, when taking pictures with a camera, skylight[a] hits an object, and the light bounced off the object, called direct light, reaches the camera to capture the image. On the other hand, if haze or mist is present at the time of shooting due to

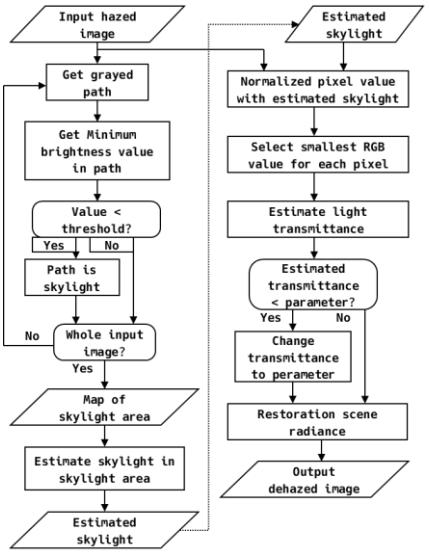


Fig. 1. Flowchart of haze removing.

weather or other factors, the skylight will be scattered by the haze, creating what is called airglow. This atmospheric light mixes with the direct light before it reaches the camera, creating a haze in the captured image.

Based on these facts, the mechanism that causes haze can be modeled by the following equation

$$I(x) = J(x)t(x) + A(1 - t(x)). \qquad (1)$$

Here, x is the entire shooting scene, J is the direct light from the object (the original clear image), A is the airglow (assumed to be constant throughout the shooting scene), and $t(0 < t \leq 1)$ is the transmittance (mixing of direct light and skylight), $A(1 - t(x)$ is airglow, and I is the captured image (hazed image).

In order to obtain the original image, the above equation is transformed into the following equation.

$$J(x) \approx \frac{I(x) - \tilde{A}(x)}{\tilde{t}(x)} + \tilde{A}(x). \qquad (2)$$

Here, $\tilde{A}$ is an estimated value of skylight, and $\tilde{t}$ is an estimated value of transmittance. By obtaining $\tilde{A}$ and $\tilde{t}$, direct light from objects, that is, the original clear image can be obtained.

2.2. *Flowchart of haze removing*

The haze removing is divided into four stages: extraction of skylight region, calculation of the skylight estimate, calculation of the transmittance, and restoration of direct light. In this section, these four processes will be explained in broad terms (Fig. 1).

The first and second steps will be described together as the skylight estimation process. Assuming that the influence of skylight in the image is reflected in the luminance, the luminance value is calculated by the RGB value of each pixel. This is then subjected to a minimization filter for each local region on the image called a patch. The minimum value for each patch is compared with the set threshold value, and if it exceeds the threshold value, the entire patch is set as the skylight region.

Since the next calculation of the skylight estimate assumes that the value of skylight in the image is constant, it is obtained by taking the average of the RGB values of the estimated skylight region.

The third stage of transmittance estimation is calculated using the following equation, which is obtained by transforming equation (1)

$$\tilde{t}(x) \approx 1 - \alpha \min_{w \in \Omega(x)} \left[\min_{C \in \{R,G,B\}} \left\{ \frac{I^C(x)}{\tilde{A}^C} \right\} \right]. \qquad (3)$$

Here, $\tilde{t}$ is the estimated transmittance and $\alpha(0 < \alpha \leq 1)$. From equation (3), the transmittance can be calculated by implementing the darkest channel local minimization filter on the normalized haze image and finding the value adjusted by α to prevent it from becoming extremely small.

In the final process of restoring scene radiance, we use equation (2) based on the estimated skylight value and transmittance. In this process, the denominator value is adjusted so that the first term on the right side of equation (2) does not become extremely large.

This is an overview of the haze removing based on the darkest channel priority method, and the software created based on this method is called pure software.

3. Haze Removing Software for HLS

3.1. *HLS (High-level synthesis)*

HLS automatically generates circuit data described by HDL (Hardware Description Language) such as Verilog from software program. While there are advantages to using HLS, such as greatly reducing the development time and the burden on developers, there are several things to keep in mind.

For example, if the software input to the HLS uses recursive functions or floating point, the circuit size of

the resulting hardware will be unnecessarily large, and the performance will be low. Therefore, we design a software program considering HLS based on the flowchart described in Section 2.2, taking the above points into account.

3.2. *Software design considering HLS*

For the skylight estimation process, which is the first and second steps of haze removing, a software program for HLS has been developed in a previous study and its effectiveness has been confirmed[3]. Briefly there are three major changes in the skylight estimation process for high level synthesis: converting patch processing to pixel processing, removal of the skylight region extraction map, and finally, pipelining of the process. The patch processing has been converted to pixel processing and the skylight region extraction map has been removed. This strategy gives a significant improvement in terms of memory access for hardware. The pipelining of processing plays a major role in improving the processing performance of the hardware being generated.

For the third step, the transmittance estimation process, we changed the method to perform the process for each pixel as in the skylight estimation process described above, as opposed to using the local minimum value filter in the conventional method. In the case of the conventional method using the local minimum filter, the banding[b] caused by this effect is repaired by applying a smoothing process called soft matching. However, this soft matching is complex and computationally expensive, so implementing it in hardware would unnecessarily increase the circuit size. In the proposed method, the transmittance is estimated for each pixel instead of the local minimum filter, and the restoration process can be performed without smoothing and with maintaining the hardware performance.

In the last step of restoring the scene radiance, the calculation process was changed from the floating-point method used in the conventional method to a fixed-point method using unsigned 32-bit integers that can maintain the minimum necessary accuracy. This calculation method was applied in the same way in the transmittance estimation process. This will reduce the circuit size of the development hardware.

Based on the above, we created a haze removing software program considering HLS (hereafter referred to as HLS software).

4. Experiments and Discussions

4.1. *Experimental environment*

The HLS software program created in Section 3.2 was input to Xilinx's high-level synthesis tool Vivado HLS 2018.3 to perform high-level synthesis. Then, the HDL program generated by high-level synthesis was loaded on ZYBO (Zynq-7000 Development Board), an FPGA board manufactured by DIGILENT, by using Vivado 2018.3, a development environment software for Xilinx FPGAs, to verify the operation. The operating frequency for each device is 2.9GHz for the PC, 650MHz for the embedded CPU on the ZYBO board (hereafter referred to as ZYBO CPU), and 100MHz for the developed hardware, and the size of the input image used in the experiment was 1024 x 768 pixels.

4.2. *Experimental method*

The performance of the HLS hardware was evaluated in terms of the processing performance improvement ratio and the runtime power improvement ratio for each of the following four patterns, (i) pure software on PC, (ii) HLS software on PC, (iii) pure software on ZYBO CPU, (iv) HLS software on ZYBO CPU.

The processing performance improvement ratio is calculated by determining the ratio of the processing execution time in each case to the time required by the HLS hardware for processing, and the runtime power improvement ratio is obtained by multiplying the power improvement ratio [c] by the processing performance improvement ratio.[4, 5]

4.3. *Experimental results and discussions*

A graph summarizing the time required for the HLS hardware and each of (i) through (iv) listed in Section 4.2 to perform the haze removing is shown in Fig. 3.

As can be seen from Figure 3, the HLS hardware outperforms its execution time for all cases except (ii). The processing time of the HLS hardware is slower for

[b] Noise caused by discontinuities in transmittance at the borders of local regions

[c] Ratio of the respective operating frequencies of the HLS hardware and the case under comparison

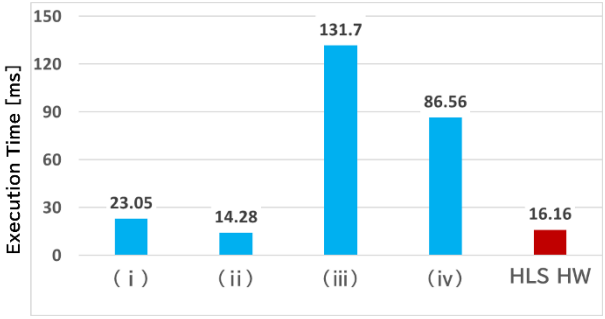


Fig. 3. Execution time of the haze removing in each case

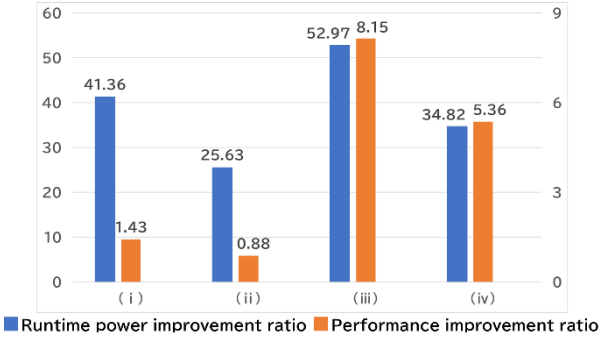


Fig. 4. Runtime power and performance improvement ratio

(ii), but this is probably due to the large difference in operating frequency.

Fig. 4 shows the runtime power and performance improvement ratios of the HLS hardware for each case.

As for the performance improvement ratio, since it is a ratio of execution time, we were able to achieve performance improvement for all cases except (ii). As for the runtime power improvement ratio, even in the case of (ii), where no performance improvement was achieved, we were able to achieve a power improvement of approximately 25.6 times.

5. Conclusion

We have developed a low-power and high-performance hardware of the haze removal process using high-level synthesis by restricting algorithm.

Although the HLS hardware could not even surpass the processing time of the software used in HLS when run on a PC, it was able to improve the runtime power with almost no loss in processing performance.

In the future, we will actually mount the hardware developed in this paper on a drone to verify its usefulness.

References

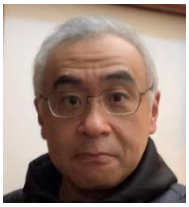
1. K. He et al: “Single Image Haze Removal Using Dark Channel Prior. Proc.”, IEEE CVPR, Vol.1, pp.1956-1963, 2009
2. Hiroaki Kotera: “A color Correction for Degraded Scenes by Air Pollution”, Journal of the Color Science Association of Japan, Vol.40, No.2, pp. 49-59, 2016
3. D. Shirai, A. Yamawaki: “Hardware Development of Skylight Estimation Processing in Haze Removing Using High-level Synthesis”, ICIAE2021, pp107-111, 2021
4. M. Yamasaki et al: “Effect of Redundant Function Execution to Reduce Memory Access on High-level Synthesis”, Processing of the 6th IIAE International Conference on Industrial Application Engineering 2018, pp.206-209, 2018
5. M. Yamasaki, A. Yamawaki: “Description Method for High-level Synthesis of Histogram Generation and Their Evaluation”, IEIE Transactions on Smart Processing and Computing, Vol. 8, No. 3, pp.178-185, 2019

Authors Introduction

Mr. Daiki Shirai

He received his bachelor’s degree from Faculty of Engineering, Department of Electrical and Electronic Engineering, electronic Systems Engineering Course, Kyushu Institute of Technology, Japan in 2020. He is currently a master’s course student in Kyushu Institute of Technology, Japan.

Dr. Akira Yamawaki



He is an Associate Professor of Faculty of Engineering at Kyushyu Institute of Technology in Japan. He received his PhD. in Electrical Engineering from the Graduate School of Engineering, Kyushu Institute of Technology, in 2006. His reserch interest in Digital Circuit Systems.