FPGA-based Implementation of the Stereo Matching Algorithm using High-Level Synthesis

Iman Firmansyah and Yoshiki Yamaguchi

Faculty of Engineering, Information, and Systems - University of Tsukuba.
1-1-1 Ten-ou-dai, Tsukuba Ibaraki, 305-8573, Japan

December 20, 2021

Outline

1. Objective of research
2. Introduction to stereo vision
3. Evaluation of stereo matching algorithm using Xilinx Zynq FPGA
4. Experimental results
5. Conclusion

Objective of research

- This study proposes a stereo vision system for FPGA implementation using high-level synthesis.
- To reduce FPGA resources especially for a small FPGA with limited hardware resources, local-block matching with small window buffer is employed for the sum-absolute difference.
- Our contribution introduces the implementation of the secondary consistency checking in the post-processing to remove the occluded pixels in the stereo matching application.

Introduction to stereo vision

- Stereo vision uses two images (left and right) for extracting the object's disparity information.
- It assesses the corresponding pixels from the two images located on the same baseline.
- Due to the capability to gather depth information, the stereo vision implementation can be found in automotive for autonomous driving applications and automatic braking, robot navigation for localization, and for crop height measurement in agriculture.

Stereo matching for generating the disparity map

- Stereo matching algorithm for this study includes SAD, consistency checking, and the generation of the disparity map.

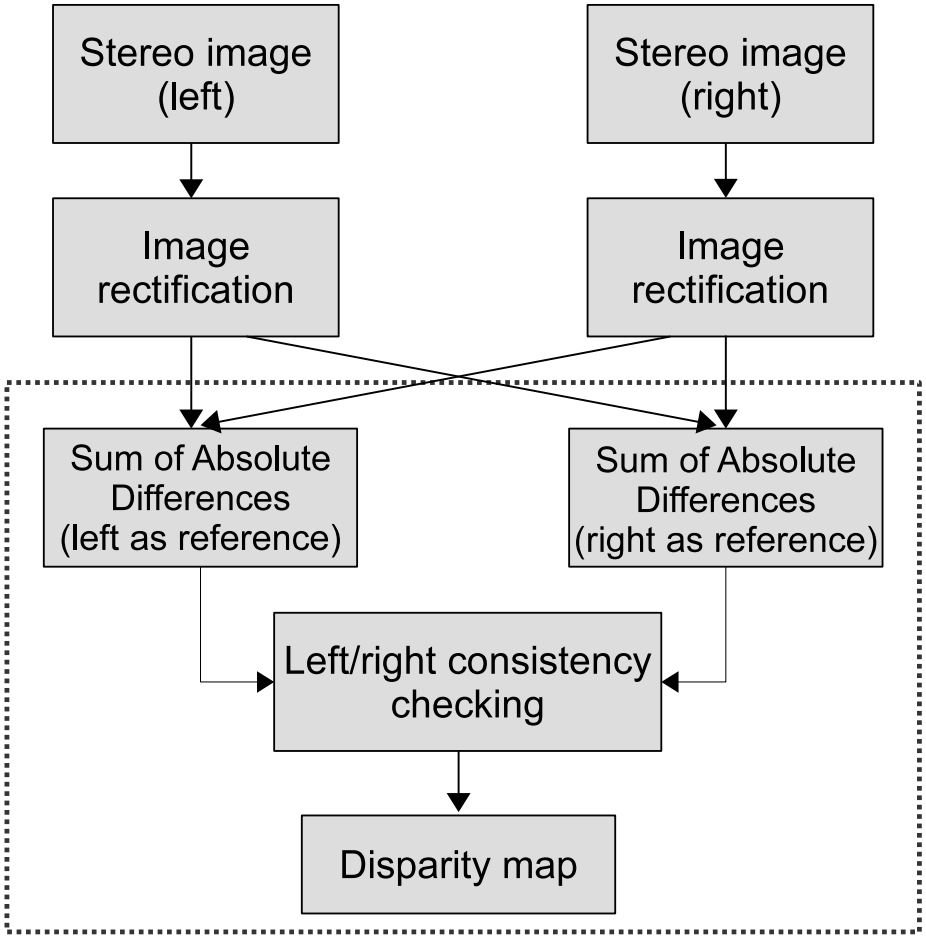

Figure 1: Stereo matching implementation using the SAD algorithm.

Sum of Absolute Difference (SAD)

- SAD is an example of algorithms widely used for finding the corresponding pixels in stereo images. SAD algorithm fetches every pixel in a block and computes the SAD from two stereo images.

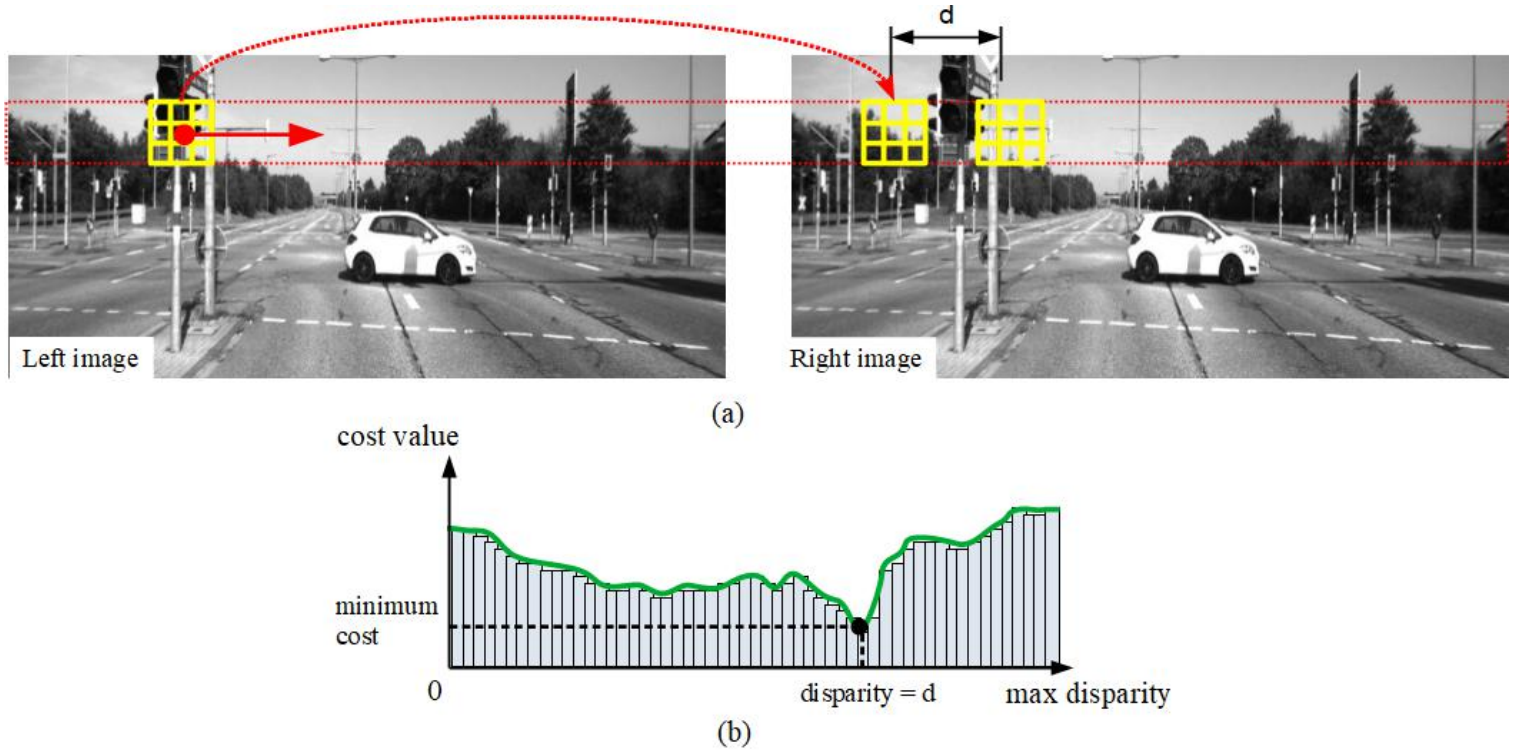

Figure 2: Program execution using SDSoC implementation.

- The matching cost for SAD is computed as represented in (1).

$$D_{(x,y,d)} = \sum |I_L(x,y) - I_R(x-d,y)| \quad (1)$$

FPGA Implementation of Stereo Matching

- SDSoC or Software-Defined System-On-Chip is employed to develop stereo matching on the Xilinx Zynq UltraScale+ MPSoC (ZCU102).

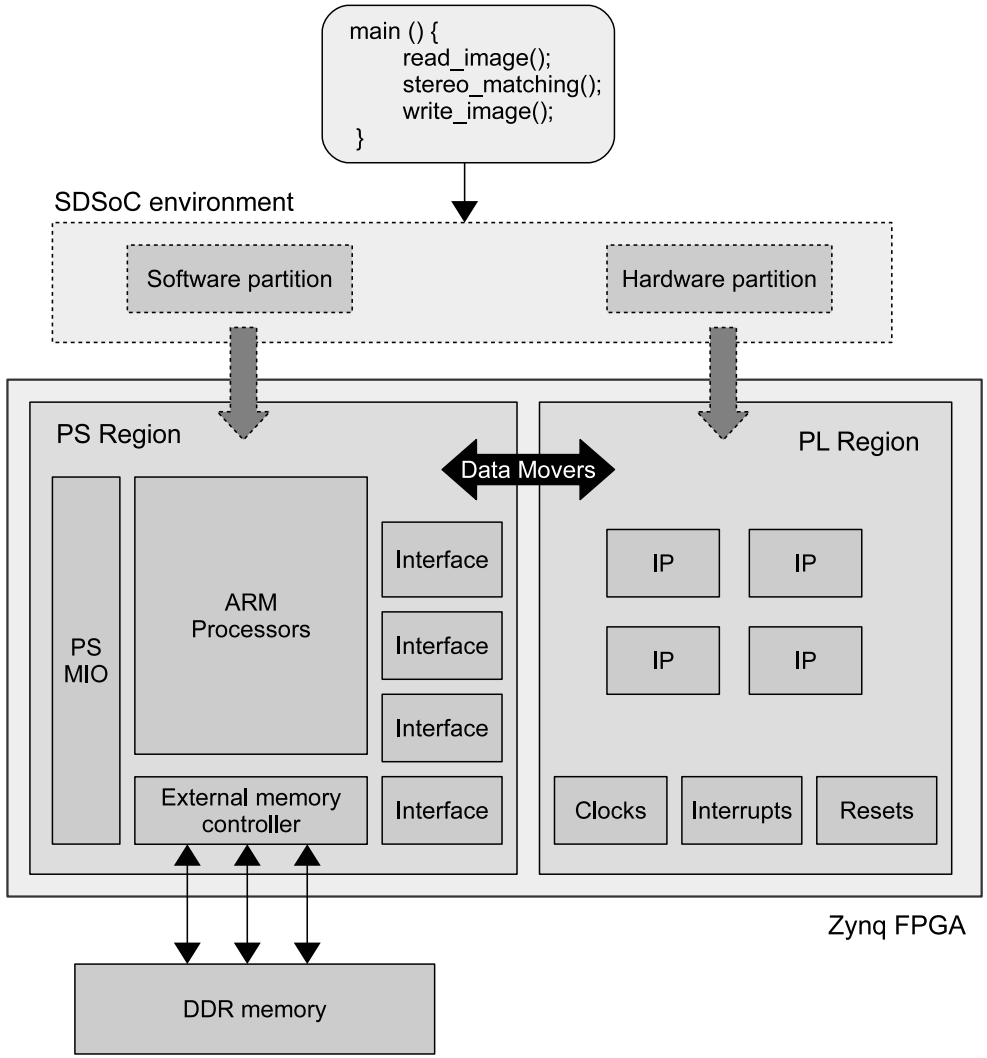

Figure 3: Xilinx Zynq FPGA development using SDSoC.

FPGA Implementation of Stereo Matching

- The CPU of PS part starts with the allocation of an array of a memory buffer that contains the image data set and for the disparity image computed by the PL part.

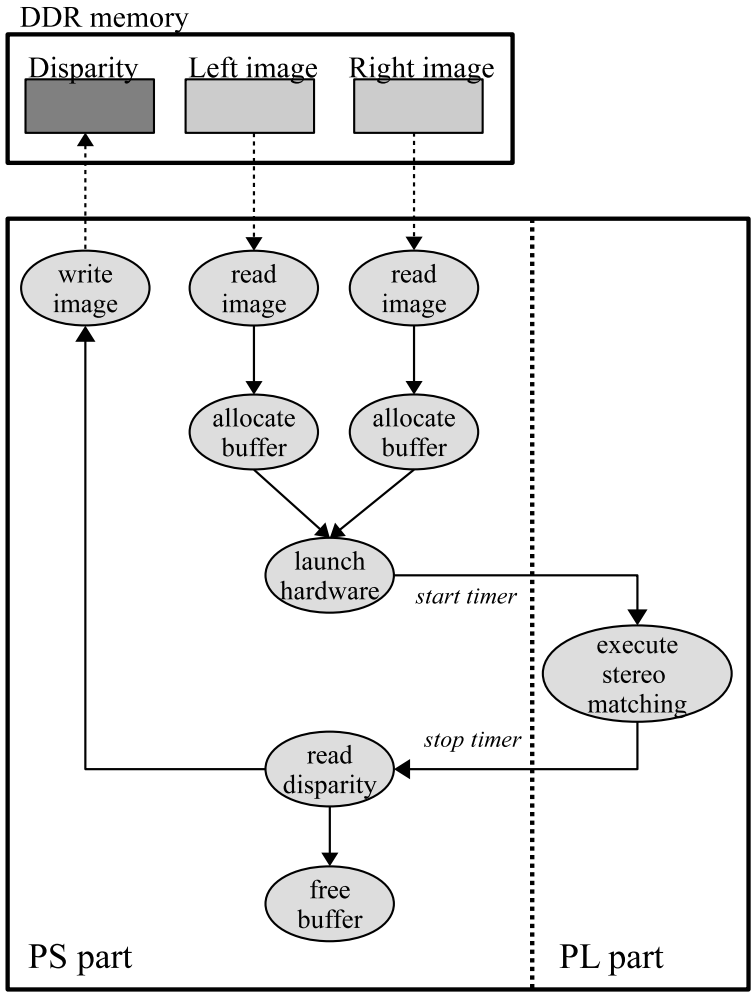

Figure 4: Program execution using SDSoC implementation.

Secondary consistency checking

- Non-occluded pixels for $D_{LR}(x, y)$:

$$|D_L(x, y) - D_R(x + D_L(x, y), y)| \leq 1 \quad (2)$$

- Meanwhile, for $D_{RL}(x, y)$, the criterion is

$$|D_R(x, y) - D_L(x - D_L(x, y), y)| \leq 1 \quad (3)$$

- The criterion for $D'_{LR}(x, y)$ is

$$|D_{LR}(x, y) - D_{RL}(x + D_{LR}(x, y), y)| \leq 1 \quad (4)$$

- Meanwhile, for $D'_{RL}(x, y)$, the criterion is

$$|D_{RL}(x, y) - D_{LR}(x - D_{RL}(x, y), y)| \leq 1 \quad (5)$$

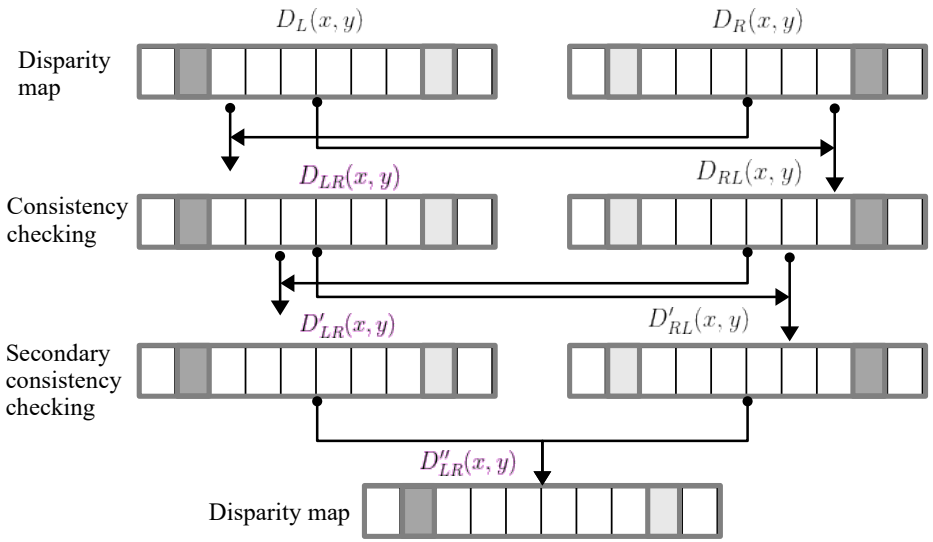

Figure 5: Disparity map using secondary consistency checking.

- The final disparity image $D''_{LR}(x, y)$ is obtained from $D'_{LR}(x, y)$ and $D'_{RL}(x, y)$

$$\left|D'_{LR}(x, y) - D'_{RL}(x + D'_{LR}(x, y), y)\right| \leq 1 \quad (6)$$

Experimental Results using KITTI dataset

Table 1: Zynq UltraScale+ MPSoC ZCU102 FPGA resource usage.

Image size	BRAM	DSP48E	FF	LUT	Freq. (MHz)	Time (s)
375x1242	38 (2%)	0	6,753 (1%)	19,260 (7%)	299	0.051

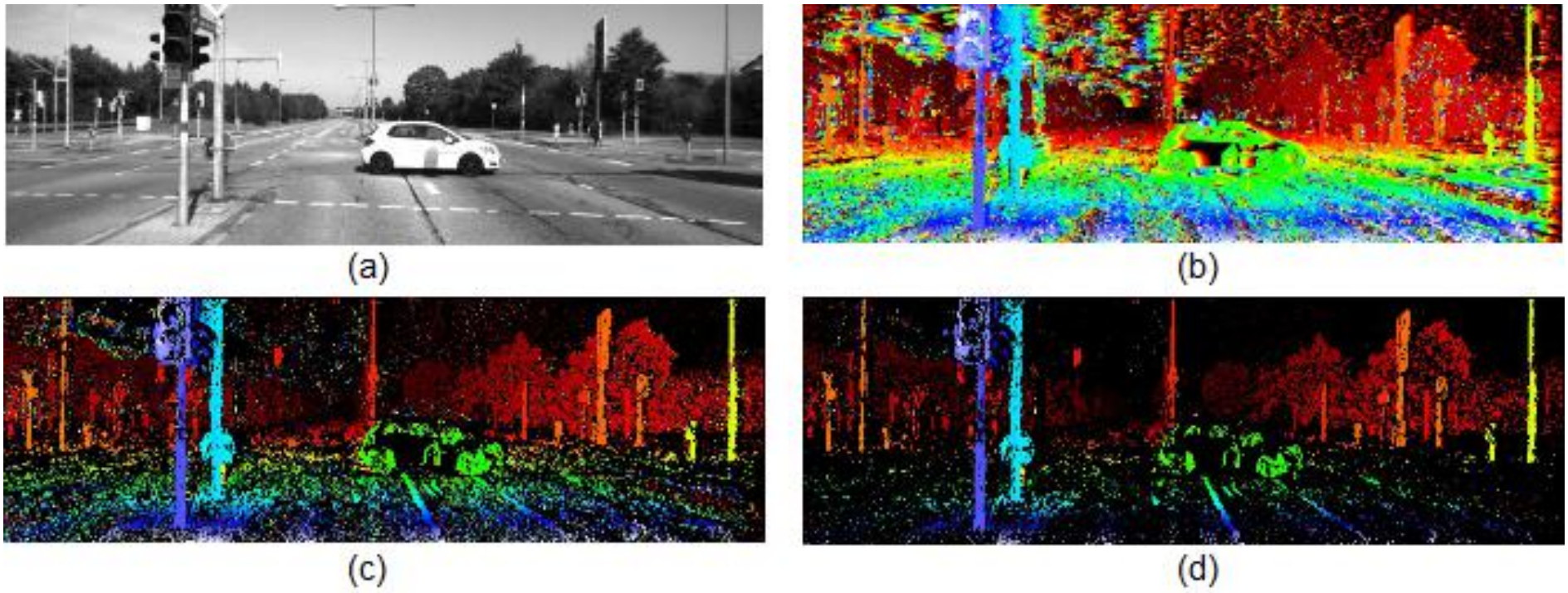

Figure 6: KITTI dataset image (a) left image for reference (b) disparity image using SAD (c) left-right consistency checking (d) secondary consistency checking.

Experimental Results using Middlebury dataset

Table 2: Zynq UltraScale+ MPSoC ZCU102 FPGA resource usage.

Image size	BRAM	DSP48E	FF	LUT	Freq. (MHz)	Time (s)
486x720	38 (2%)	0	4,981 (1%)	19,303 (7%)	299	0.038

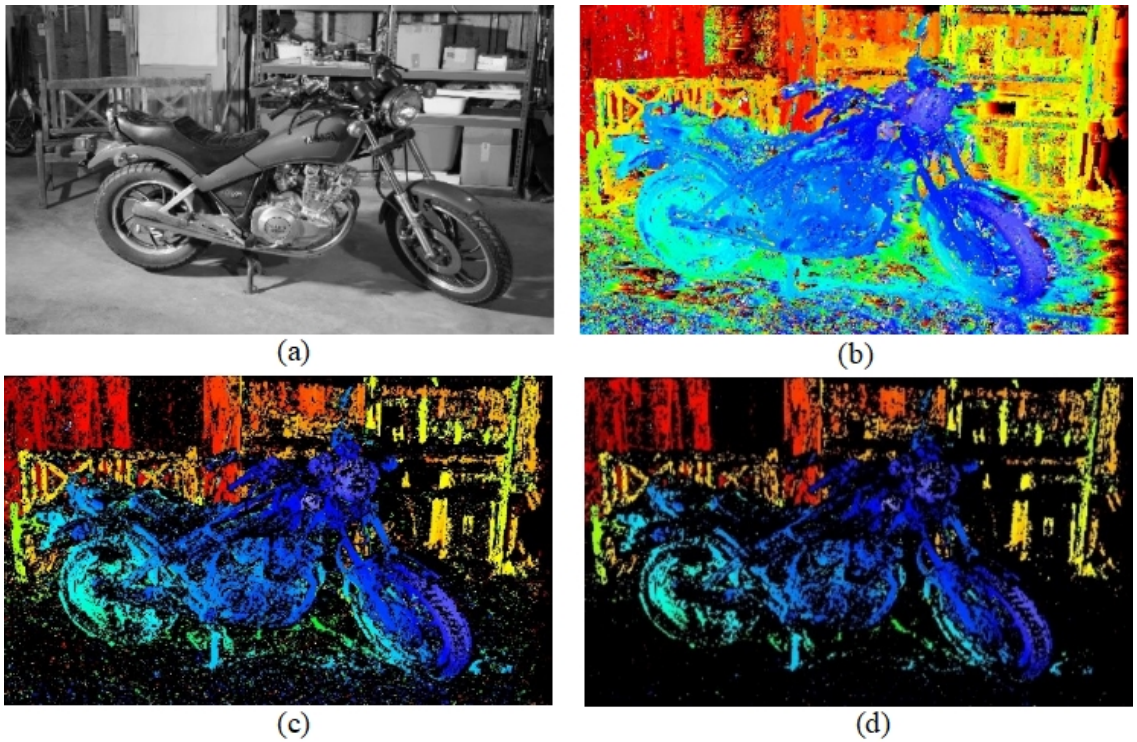

Figure 7: Middlebury dataset image (a) left image for reference (b) disparity image using SAD (c) left-right consistency checking (d) secondary consistency checking.

Performance comparison

- Performance comparison was evaluated using a single thread of ARM Cortex-A53 with a CPU frequency of up to 1.5 GHz.

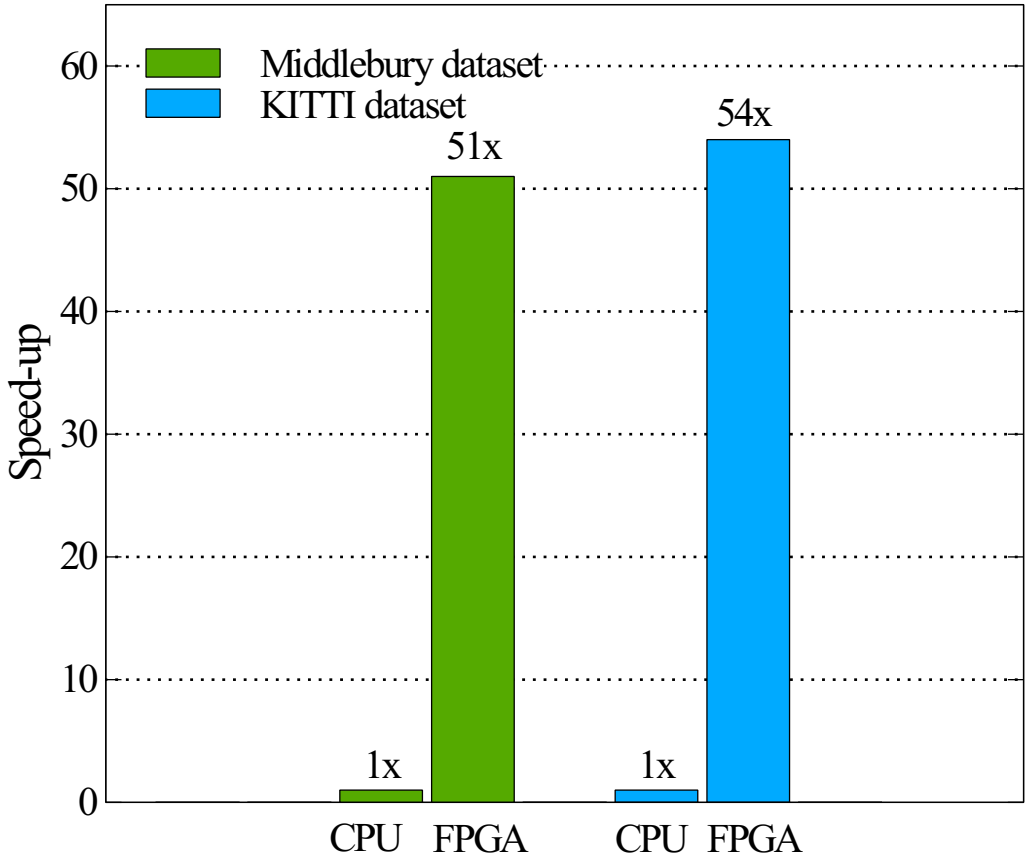

Figure 8: Performance comparison between ARM CPU (PS part) and FPGA (PL part).

Conclusion

- This study demonstrated an FPGA-based stereo matching implementation for stereo vision using high-level synthesis.
- For reducing the occluded pixels in the disparity map affected by the implementation of a small window buffer, we applied a secondary consistency checking in post-processing.
- From the experimental result, it is found that the proposed design needs up to 1% of BRAM, 1% of FF, 7% of LUT, and the processing speed of 0.038 s and 0.051 s for image sizes 486x720 pixels and 375x1242 pixels, respectively, using 299 MHz.

Thank you...