INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11 MPEG2017/ M40657
April 2017, Hobart, Australia

Source	**NICT, TDU, PUT**
Status	**Report**
Title	**View Synthesis Reference Software (VSRS) 4.2 with improved inpainting and hole filing**
Author	**Takanori Senoh (NICT, t.senoh@nict.go.jp)** **Kenji Yamamoto (NICT, k.yamamoto@nict.go.jp)** **Nobuji Tetsutani (TDU, tetsutani@im.dendai.ac.jp)** **Hiroshi Yasuda (TDU, mpegyasuda@mail.dendai.ac.jp)** **Krzysztof Wegner (PUT, kwegner@multimedia.edu.pl)**

Abstract

This document shortly describes an improved common hole inpainting and hole edge filter, added to the View Synthesis Reference Software (VSRS) 4.2 as released on April 2017. The first version of the tool has been presented by NICT and TDU (m38979 [1]) on Chengdu meeting and has been harmonized with other tools in VSRS by joint works of NICT, TDU and Poznań University of Technology.

1 Introduction

According to the decision made at the 117th MPEG meeting, the improved common hole inpainting tool and hole edge filter, which were reported in M38979 (iVSRS), were adopted to View Synthesis Reference Software as version 4.2. New configuration parameter "IvsrsInpaint" have been added to switch on/off the included common hole inpainting tool together with the hole edge filtering. New tools have been harmonized with all other tools already present in VSRS and work with both 8-bit and 16-bit depth maps.

(a) VSRS4.1 (b) VSRS 4.2

Figure 1. Comparison of VSRS 4.1 vs VSRS 4.2
with improved common hole impaint and hole edge filter

2 Improved hole inpainting process

Previous version of VSRS fills holes in synthesized view from the left (or the right) view with data from the right (or the left) view. After that, it merges the left and the right synthesized views together and then inpaints common holes with their surrounding pixels unconditionally. This process leads to two problems:

1. Hole edge error (skinny ghost)
   Since the combination of forward depth warping and then backward texture warping can generate one pixel uncertainty at edges of the object, texture edge and depth edge don't completely match to each other. In Figure 1, a foreground object edge is in the air, which is caused by this problem.
2. Inpaint noise (fat ghost)
   Since holes are generated after foreground objects are displaced, where background objects must appear. If such holes are unconditionally inpainted with the surrounding pixels, holes may be inpainted with the foreground color as shown in Figure 1.

These abovementioned problems have been solved in the following improved inpainting process. Steps 1-2 are designed to solve the skinny ghost problem, while steps 3-8 solve the fat ghost problem. Common holes are inpainted with the background pixels selected from 16 directions.

Step1: Dilate holes in synthesized view from the left (or the right) view.
Step2: Fill dilated holes with synthesized view from the right (or the left) view.

Step3: Merge synthesized left view and right view and make a common hole mask.
Step4: Find left and right edge pixels of a common hole, line by line.
Step5: Search a hole edge pixel of the smallest depth value, in 16 directions from the center of each one-line hole.
Step6: Fill the one-line hole with the pixel having the smallest depth value.
Step7: After all holes are filled, erode the hole by one pixel.
Step8: Inpaint eroded hole.

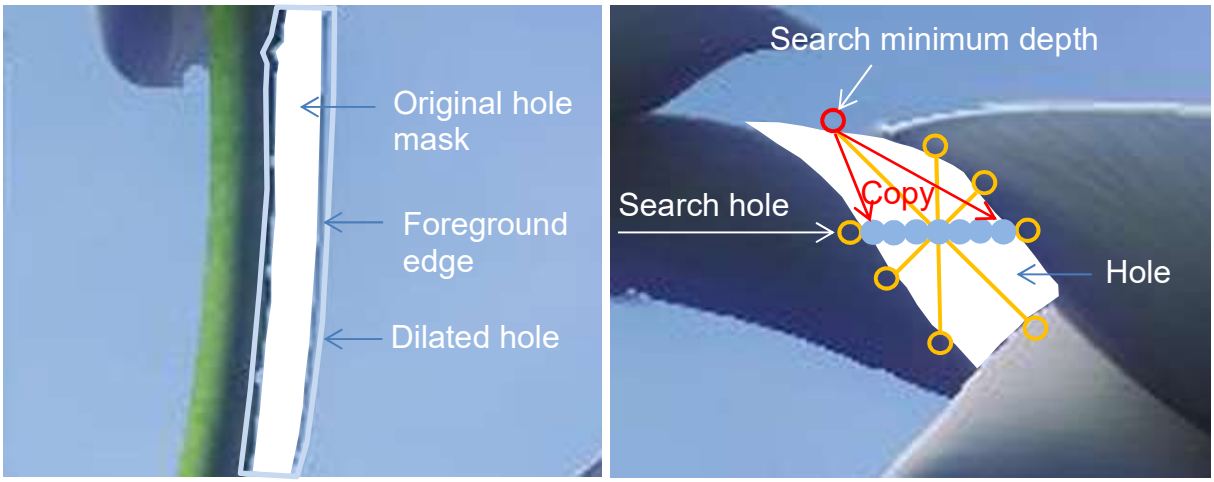


Figure 2. Hole dilation (left) and hole filling with background pixel (right)

Figure 3 shows exemplary view of Flowers dataset synthesized by the VSRS 4.2. Both skinny ghost and fat ghost are reduced.

Figure 3. View no. 24 synthesized from view no. 19 and view no. 32 by VSRS 4.2

3 Hole edge filter

When color profile of left and right views are different as shown in Figure 4a hole edges problem occurs. In order to minimize this problem filtering of the hole edge is applied if both side of such edge belongs to the same object i.e. there is no corresponding depth edge. Filtering operation are explained by following pseudo-code

```
if(hole_edge[a] && no_depth_edge[a]) {
  pel[a] = (pel[a-2]+pel[a+1])/2;
  pel[a-1] = (pel[a-2]+pel[a])/2;
  pel[a+1] = (pel[a]+pel[a+1])/2; }
```

(a) VSRS 4.1 (b) VSRS 4.2

Figure 4. Standing out hole edge caused by the color difference between left and right views

4 New configuration parameters

The integration of improved common hole inpainting tool and hole edge filter into VSRS imposed introduction of additional configuration parameter to configuration file. The following parameter has been added to configuration files:

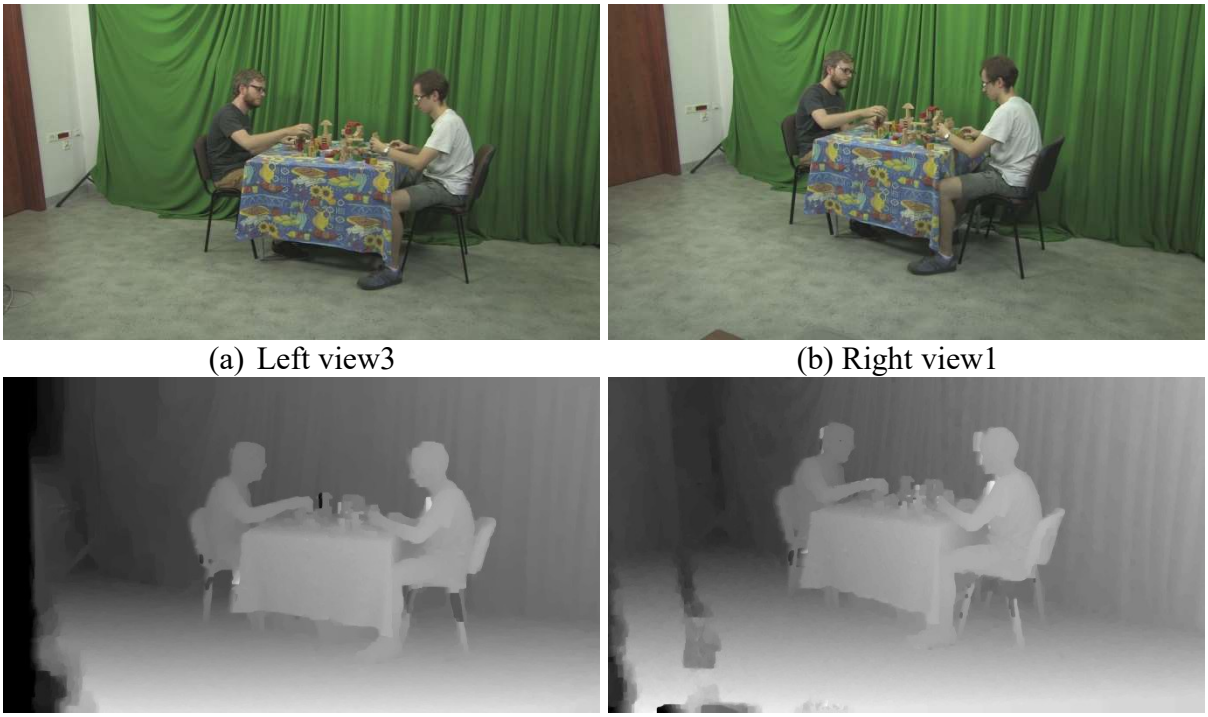
```
IvsrsInpaint    1 # 0...Conventional, 1...Improved inpainting
```

The second column presents typical parameter values and the last column presents description of parameters meaning.

5 Qualify assessment and Verification

5.1 Verification with Poznan_Blocks2

For 16-bit depth map verification, Poznan_Blocks2 sequence was used. Figure 5 shows view no. 1 and view no. 3 used for the verification. Since 16-bit depth maps were difficult to download caused by the server access trouble, they were estimated by DERS 6.0 with the attached Ders16_Blocks2.cfg file. Figure 2 also shows the 8-bit truncated 16-bit depth maps, since the 16-bit depth maps were difficult to show directly.



(a) Left view3 (b) Right view1

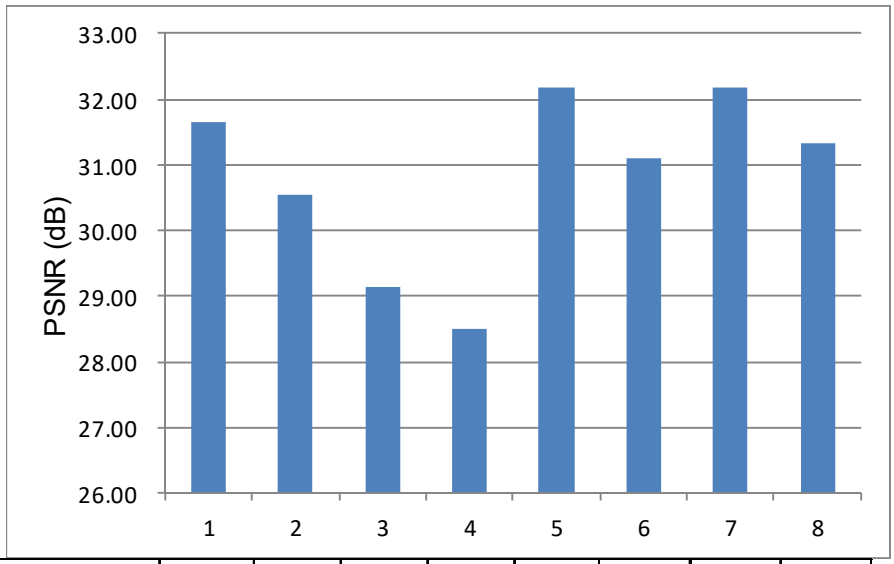
(b) Left depth3 (b) Right depth1

Figure 5. Poznan_Blocks2 used for VSRS4.2 verification.

Figure 6 shows the best synthesized view no. 2 (32.19dB) with ViewBlending=1 (not blend), DepthBlendDifference=65535 (always blend), IvsrsInpaint=0 (not use). The cfg file used for the VSRS 4.2 is also attached to this contribution.

Figure 6. Synthesized view no. 2 (32.19dB) with ViewBlending=1 (not blend), DepthBlendDifference=65535 (always blend), IvsrsInpaint=0 (not use)

Figure 6 shows the PSNR of synthesized view no. 2 for several parameter combinations. DepthBlendDifference=65535 means to blend views for all depth differences. For Poznan_Blocks2, only ViewBlending=1 (not blend) is useful but the other tools are not useful. When Boundary Noise Removal is ON (BoundaryNoiseRemoval=1), the PSNR dropped by 0.8 dB.

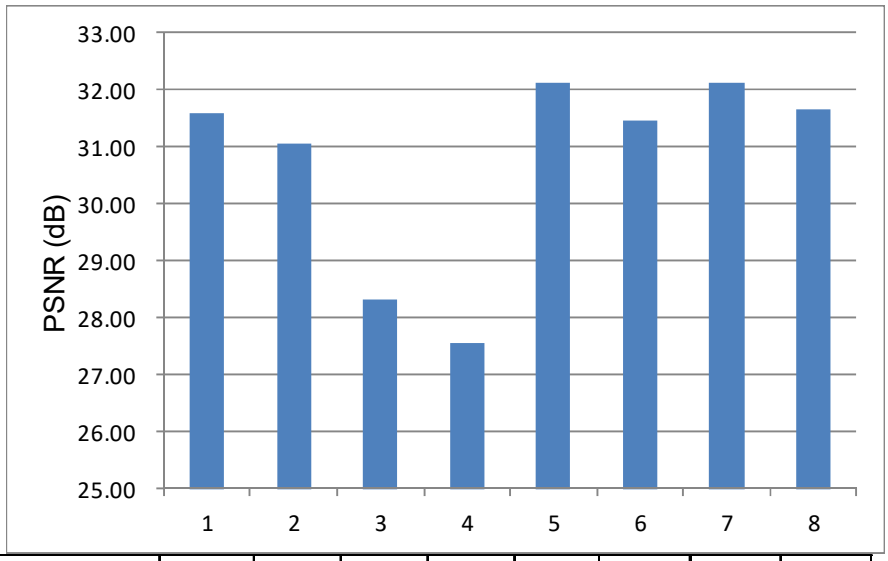


ViewBlending	0	0	0	0	1	1	1	1
DepthBlendDifference	65535		5	5	65535		5	5
IvsrsInpaint	0	1	0	1	0	1	0	1

Figure 7. PSNR of Synthesized view2 from 16-bit depth maps.

For your reference, Figure 5 shows PSNRs of synthesized view2 from the truncated 8-bit depth maps. VSRS4.2 can be switched to 8-bit depth mode by commenting out “#define

POZNAN_16BIT_DEPTH" in version.h file in \ViewSynLibStatic\include. Some performance degradation from the 16-bit depth map was observed.



ViewBlending	0	0	0	0	1	1	1	1
DepthBlendDifference	65535		5	5	65535		5	5
IvsrsInpaint	0	1	0	1	0	1	0	1

Figure 5. PSNR of Synthesized view2 from 8-bit depth maps.

6 Verification with BBB_Flowers_NoBlur

For 8-bit depth map verification, BBB_Flowers_NoBlur was also used. Figure 6 shows the views and depth maps used for the verification.

(c) Left view25 (b) Right view32

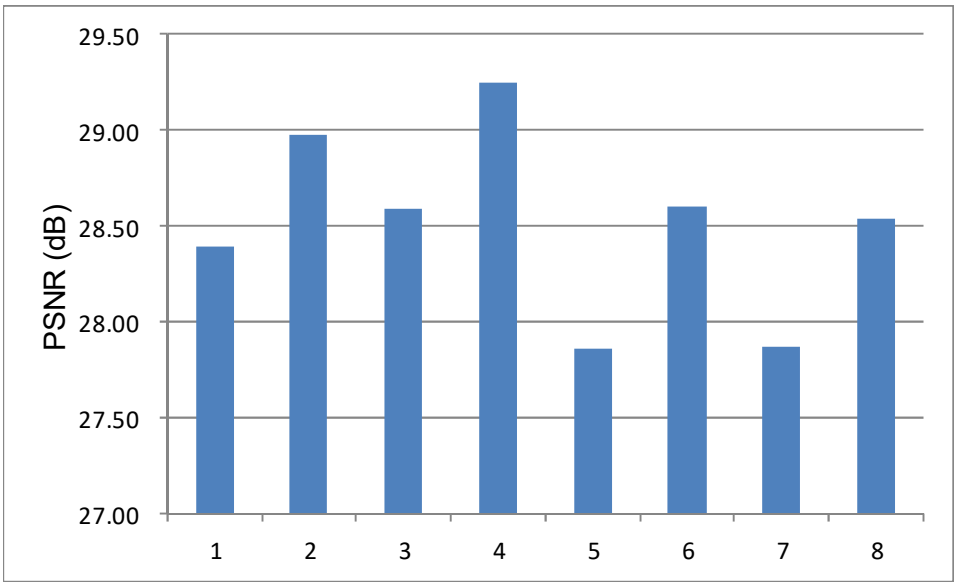
(c) Left depth25 (d) Right depth32

Figure 6. BBB_Flowers_NoBlur used for VSRS4.2 verification.

Figure 7 shows the best synthesized view28 (29.25dB) with ViewBlending=0 (blend), DepthBlendDifference=5 (blend if <5), IvsrsInpaint=1 (use). Its cfg file is also attached.

Figure 7. Synthesized view28 (29.25dB) with ViewBlending=0 (blend), DepthBlendDifference=5 (blend if <5), IvsrsInpaint=1 (use).

Figure 8 shows the PSNR of synthesized view28 for several parameter combinations. For Flowers_NoBlur, Depth-base View Blending (DepthBlendDifference=5) and improved common hole inpainting together with hole edge filter (IvsrsInpaint=1) are useful but View Blend Inhibit (ViewBlending=1) is not useful. This combination is completely opposite to the Blocks2 combination. When Boundary Noise Removal is ON (BoundaryNoiseRemoval=1), the PSNR dropped by 2.5 dB.



ViewBlending	0	0	0	0	1	1	1	1
DepthBlendDifference	65535		5	5	65535		5	5
IvsrsInpaint	0	1	0	1	0	1	0	1

Figure 8. PSNR of Synthesized view28 of Flowers_NoBlur.

7 Conclusion

A new version of View Synthesis Reference Software (VSRS 4.2) with improved common hole inpainting and hole edge filter have been released. Correctness of new release under various configuration have been tested and verified. New release can be found on MPEG SVN.

8 References

[1] Takanori Senoh, Kenji Yamamoto, Nobuji, Hiroshi Yasuda, “VSRS improvement”, ISO/IEC JTC1/SC29/WG11 MPEG2016/ M38979, October 2016, Chengdu, China

[2] Krzysztof Wegner, Olgierd Stankiewicz, Masayuki Tanimoto, Marek Domanski „Enhanced View Synthesis Reference Software (VSRS) for Free-viewpoint Television” ISO/IEC JTC1/SC29/WG11 MPEG2013/M31520, October 2013, Geneva, Switzerland

[3] Krzysztof Wegner, Olgierd Stankiewicz, Marek Domański, “Depth based view blending in View Synthesis Reference Software (VSRS)”, ISO/IEC JTC1/SC29/WG11 MPEG2015/M37232, October 2015, Geneva, Switzerland