Real-time free viewpoint rendering via view-dependent polygon plane arrangement

Keisuke Nonaka, Ryosuke Watanabe, Jun Chen, Sei Naito
KDDI Research, Inc.

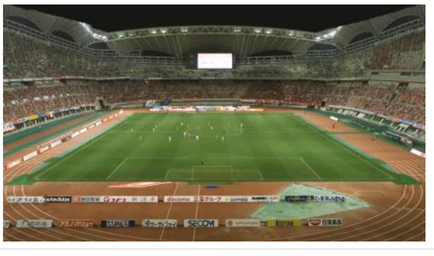
Background (What is free-viewpoint?)

- **Free-viewpoint synthesis (Free-viewpoint video)**
  - **To synthesize virtual image from arbitrary viewpoint by using multiple images**
    - Users can watch a scene from arbitrary angle freely
  - **Feature: immersive experience (has high affinity to virtual reality experience), Intuitive understanding of the target scene**
  - **Use case: sports watching, training, analysis, education, gaming, etc.**

2

Background (Example)

- **An example of free-view point system (KDDI, 2012)**



3

Background (Approaches)

- **How do we synthesize the video?**
  - **Several approaches have been proposed.**
  - **Image based**
    - Interpolation of multiple images
    - Stereo matching (considering parallax)
    - Camera lenses array (based on light field theory)
  - **Model based (Depth based)**
    - Visual hull (point cloud)
    - Depth sensor based

4

Background (Comparison)

- **Comparison of features with several approaches**

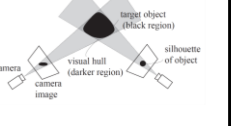
	Virtual viewpoint range	Calculation cost	Distance between multiple cameras	Restriction of equipment
Interpolation	Narrow	Low	Narrow	Only camera
Stereo matching	Narrow	Low	Narrow	Only camera
Camera lenses array	Narrow	High	Narrow	Light field camera
Visual hull	Wide	High	Wide	Only camera
Depth sensor based	Wide	Low	Wide	Infrared camera, etc.

- **We chose "visual hull" as an approach to realize the free-viewpoint video considering these advantages for customer use.**
  - **Wide viewpoint range (sparsely arranged cameras)**
  - **High quality synthesis**
  - **not require special equipment**

5

Conventional method of the visual hull

- **Standard concept of conventional method [1]**
  - **Calculate 3D (three dimensional) shape of a object (e.g. human) by using an intersection of silhouette in each image**
    - Silhouette of object represents a cone shape in 3D space
  - **The shape is represented as a 3DCG by a set of many vertices**



[1] Hiroshi Sankoh, Sei Naito, Keisuke Nonaka, Houari Sabirin, and Jun Chen. Robust Billboard-based, Free-viewpoint Video Synthesis Algorithm to Overcome Occlusions under Challenging Outdoor Sport Scenes. In *Proceedings of the 26th ACM international conference on Multimedia* (MM '18). ACM, pp. 1724-1732, 2018.

6

Conventional method of the visual hull

- **Problem**
  - **It takes so much time to calculate the shape even by using decent power PC**
    - It is quite impactful for realizing real-time live streaming of free-viewpoint video
- **Proposal**
  - **We propose a new concept of representation of the visual hull to overcome the above problem**
    - use a set of virtual planes instead of a set of voxels

The outline of conventional method

Data aquisition → Mask extraction → Voxel setting → Marching cube → Texture projection

Totally the same in proposed method

The detail of conventional method (1/4)

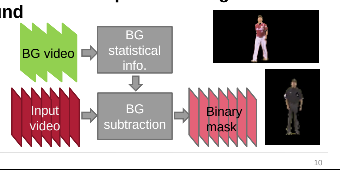
Data aquisition → Mask extraction → Voxel setting → Marching cube → Texture projection

- **Multiple camera images** $Si(i \in \{1, . . .,K\})$ **are acquired**
  - **K represents the number of cameras**
- **Some conditions are manually (or automatically) adjusted in advance**
  - **Time synchronization**
  - **Camera parameter calculation (intrinsic and extrinsic)**

The detail of conventional method (2/4)

Data aquisition → **Mask extraction** → Voxel setting → Marching cube → Texture projection

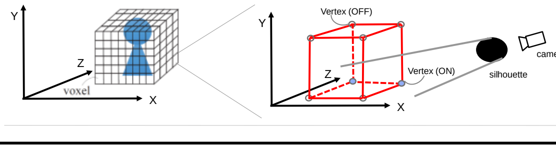
- **We adopt simple background (BG) subtraction method to obtain a silhouette of object**
  - **BG video with no target object is shot in advance**
  - **BG subtraction on certain threshold separate a image into foreground and background**



The detail of conventional method (3/4)

Data aquisition → Mask extraction → **Voxel setting** → Marching cube → Texture projection

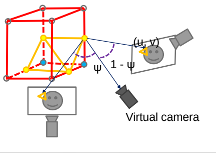
- **In the virtual 3D space, the method set a virtual 3D rectangular parallelepiped (voxel) along XYZ axis**
  - **Each voxel has 8 vertices which share the neighbor voxel**
  - **By projecting the silhouette mask onto a plane, a flag (ON or OFF) of a vertex whether it is included in object shape can be judged**
  - **A set of vertices which represent object shape would be obtained as point cloud**



The detail of conventional method (4/4)

Data aquisition → Mask extraction → Voxel setting → **Marching cube** → **Texture projection**

- **For converting the vertex data to a plane (polygon), a well known scheme "Marching cube [2]" is conducted**
- **For natural synthesis, texture of the polygon is obtained by blending multiple camera images according to the distance ψ from a virtual viewpoint to the cameras**
- **Finally, 3DCG data of object is obtained**



[2] William E. Lorensen, Harvey E. Cline: *Marching Cubes: A high resolution 3D surface construction algorithm.* In: *Computer Graphics*, Vol. 21, Nr. 4, July 1987

Problem of conventional method (repeated)

- The conventional visual hull takes so much time to calculate the shape even by using decent power PC
- Ex: for representing 10 m^3 space at 1 cm^3 voxel size, we have to calculate 10^9 vertices ON/OFF flag explicitly.

13

Proposed method

- We propose a new concept of representation of the visual hull based on virtual plane
  - It economizes the time to calculate shape of the target object
  - It constructs almost the same shape compared with conventional visual hull (based on voxel)
  - It synthesizes a virtual image (video) from arbitrary virtual viewpoint, maintaining quality of conventional visual hull
- Key idea
  - We use a set of virtual plane instead of a set of voxels

14

The outline of proposed method

Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

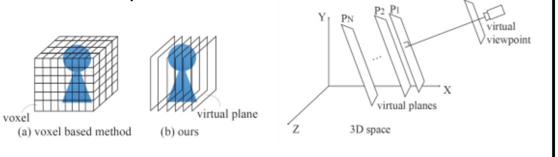
Totally the same as the conventional method

15

The detail of proposed method (1/5)

Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

- We use plane-wise expression of the shape
  - P_n: Plane with index n
  - N: the number of plane



16

The detail of proposed method (2/5)

Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

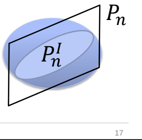
- A back-projected silhouette mask on a plane is calculated as follows:

$$P_n(x, y, z) \propto H_i M_i(u, v, 1),$$

H_i: (extrinsic) camera parameter
M_i: silhouette binary mask image

- An intersection of all the mask P_n^I can be calculated as,

$$P_n^I(x, y, z) = \prod_{\forall(u,v)\in M_i} P_n(x, y, z).$$



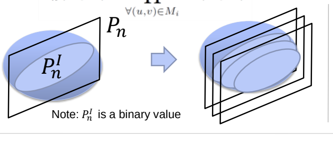
17

The detail of proposed method (3/5)

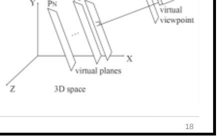
Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

- This P_n^I represents a section of target object
  - Emulating a set of vertices along the P_n (left figure)
  - To avoid virtual viewpoint dependency, the planes face toward the virtual viewpoint perpendicularly (right figure)

$$P_n^I(x, y, z) = \prod_{\forall(u,v)\in M_i} P_n(x, y, z).$$



Note: P_n^I is a binary value



18

The detail of proposed method (4/5)

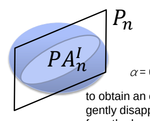
Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

- The binary shape representation is too strict considering the noise of the mask
  - Opacity of the shape, A_i, is adopted to avoid the noise
  - the parameter α should be less than 1 (in this proposal, we set α = 0.01)

$$A_i(u,v) = \begin{cases} 1 & M_i(u,v) = 1 \text{: pixel in the target object} \\ \alpha & otherwise. \text{ : pixel not in the target object} \end{cases}$$

$$PA_n(x,y,z) \propto H_i A_i(u,v)$$

$$PA_n^I(x,y,z) = \prod_{\forall (u,v) \in M_i} P_n(x,y,z)$$



19

The detail of proposed method (5/5)

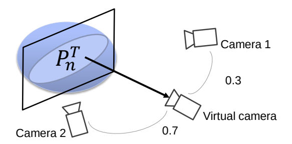
Data aquisition → Mask extraction → Virtual plane setting → Alpha projection → Texture projection

- To render a virtual viewpoint, we have to know the color of the target object
  - The difference with alpha projection is to use only Q number of camera images near the virtual viewpoint

$$P_n^T(x,y,z) = \prod_{\forall (u,v) \in S_q} \lambda_q P_n(x,y,z),$$

$$\lambda_q = \frac{\sum_{x \in Q' \setminus q} d(x)}{(Q-1)\sum_{x \in Q'} d(x)},$$

q: the index corresponding to the near Q cameras
λ_i: the weight of texture alpha blending

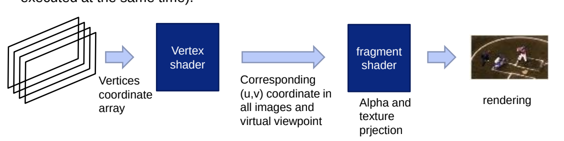


$$colorP_n^T = 0.7\ cam1 + 0.3\ cam2$$

20

Implementation

- The proposed method is consist of a combination of alpha and texture projections.
  - All the projections are proceeded easily and quickly in GPU
    - the rendered pixel value is independent for the neighboring pixel values and the current GPU is optimized for the pixel-wise procedure (shape acquisition and rendering executed at the same time).



- Disadvantage
  - Difficult to generate a general 3DCG (surface polygon) data

21

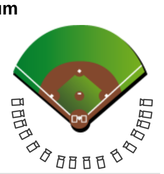
Experiments

- We carry out two experiments by using actual sports videos
  - Quality assessment
  - Comparison of calculation time

22

Experiments

- Shooting environment
  - The number of cameras: 16
  - Resolution of each camera: 1920 x 1080 (30 fps)
    - Only for the background subtraction, the input images are resized to 640 x 360
  - Data format: YUV 4:2:0
  - Target scene: baseball batting scene in stadium
  - Camera position: semicircular
- PC environment of calculation
  - OS: Windows 10 64bit
  - CPU: Intel Core i7-6700K CPU @4.00GHz
  - GPU: NVIDIA GeForce GTX 1080
  - RAM: 32GB



23

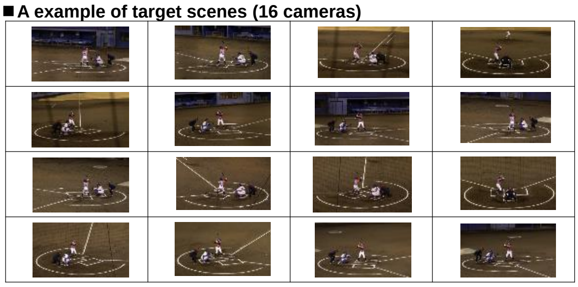
Experiments

- The other settings
  - Camera calibration: done manually in advance
    - Use geometric information of white line in batter box
  - Time synchronization of multiple cameras: manually adjusted (does not use Generator Lock)
  - Resolution of the virtual space (manually decided considering target scene):
    - Voxel [1] (conventional method): 1.0 cm^3 per each voxel
    - Ours: 1.0 cm as a distance between each plane

24

Experiments

■ A example of target scenes (16 cameras)



25

Quality assessment

■ Example of several synthesized frames

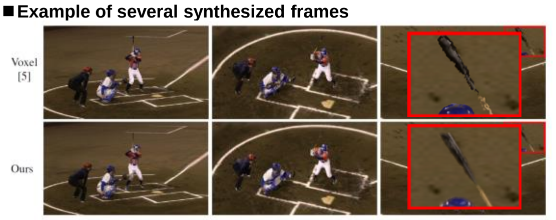
(a) Virtual Viewpoint 1 (b) Virtual Viewpoint 2 (c) Virtual Viewpoint 3

■ Some boundary artifacts have been disappeared (because of the blur effect by alpha projection)

26

Quality assessment

■ Example of several synthesized frames



(a) Virtual Viewpoint 1 (b) Virtual Viewpoint 2 (c) Virtual Viewpoint 3

■ Some boundary artifacts have been disappeared (because of the blur effect by alpha projection)

27

Quality assessment

■ A well-known structure evaluation method related to human perception, Structural SIMilarity (SSIM) [3] is used
- Ground truth: camera image
- Synthesized image: the ground truth image was not used for the synthesis (we use only 15 cameras)

	Voxel [1]	Ours
Virtual viewpoint 3	**0.686**	0.684
Average score of 10 viewpoints	**0.670**	0.669

[3] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

28

Comparison of calculation time (1/2)

■ We calculate processing time from image input to rendering (including background subtraction)
- The vertical synchronization of display has been turned off to obtain exact processing time less than refresh rate of the display
- The voxel size and the distance between each plane are the same as the previous experiment

■ Note: only for the background subtraction, the input images are resized to 640 × 360 to accelerate the processing time

29

Comparison of calculation time (2/2)

■ The time from input to rendering is described below (milliseconds order):
- Runtime: averaged rendering time measured for 5 minutes

	Voxel [1]	Ours
runtime [milliseconds]	196	**31.7 (< 33.3)**

- The background subtraction works on CPU in both method
- All the projections in our method and rendering in both method are implemented by OpenGL [4]
- The other functions in conventional voxel method [1] is implemented by CUDA for fair evaluation

[4] (2018) OpenGL. [Online]. Available: https://www.opengl.org

30

Experiments

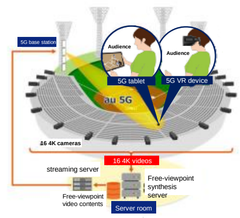
■ Please watch a synthesized video

Voxel [1] Ours

31

Experiments (Introduction of demo experiment)

■ Live-streaming experiments
- We carry out an experiment streaming the free-viewpoint video to tablet devises in stadium



32

Conclusion

■ We proposed the virtual lined plane approach to render a target object for the free-viewpoint video live streaming

■ Two experiments show that our method well represents the visual hull and it woks very quickly
- Quality: Almost same as conventional visual hull (point cloud)
- Processing time: working in real-time (less than 33msec/frame) from input to rendering

■ Future work
- Fast high accuracy mask extraction
- Content adaptive plane positioning method for economize memory use
  - To apply our method more huge target scene

33

Thank you!

34