Sampling Based Scene-Space Video Processing

Felix Klose, Oliver Wang, Jean-Charles Bazin, Marcus Magnor, Alexander Sorkine-Hornung

Overview

- Scene space video processing: pixels are processed according to their 3D positions
- What is scene space?
- Why is scene space processing advantageous?

Challenge

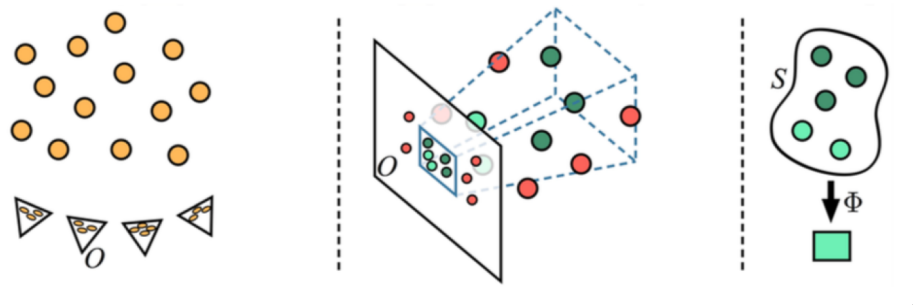
Visual output quality depends on quality of scene space information.

Idea

Do scene-space video processing on *casually captured* input video by using sample based framework instead of full 3D reconstruction.

Approach

Goal: compute all output pixel colors, $O^f(p)$.



Approach

For each $\mathbf{O}^f(p)$, draw a set of samples $\mathbf{S}^f(p)$ from $\mathbf{I}$.

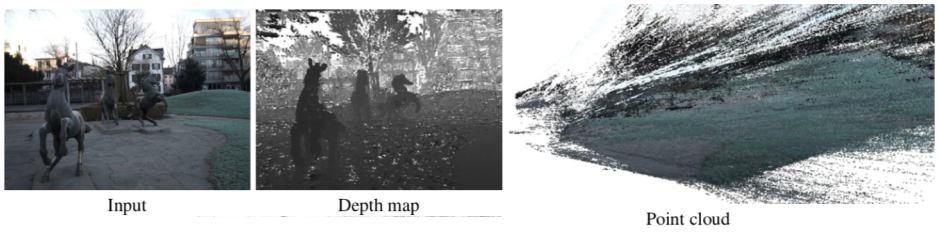
$s \in \mathbb{R}^7$ | r | g | b | x | y | z | f |

Filtering: $\Phi(S) \in \mathcal{P}(\mathbb{R}^7) \rightarrow \mathbb{R}^3$

Preprocessing

1. Use commercially available tools to compute camera calibration parameters for each input frame.

2. Use simple multi view stereo algorithm or Kinect sensor to derive dense depth information.



Input Depth map Point cloud

Step 1: Sample Gathering

- **Goal:** collect multiple observations of the same *scene point* visible to an output pixel, p.

- What is the straightforward way to do this?

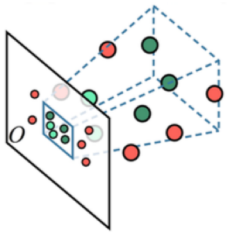
- Why not do this?

$$\underbrace{\frac{60\ sec}{1\ min}}_{\text{1 min. long}} \times \underbrace{\frac{30\ frames}{1\ sec}}_{\text{30 fps}} \times \underbrace{\frac{960 \times 720\ pixels}{1\ frame}}_{\text{720p}} = \underbrace{1{,}244{,}160{,}000}_{\text{total samples}}$$

Step 1: Sample Gathering.

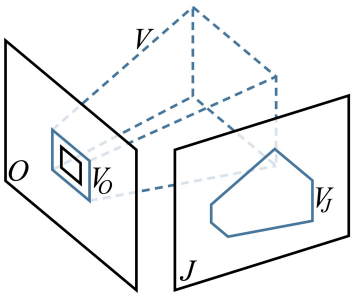
For each output pixel p, a physical camera integrates information over a frustum-shaped 3D volume V in scene-space.

$$\boldsymbol{V} = \{\boldsymbol{C_O^{-1}} \cdot [p_x \pm \frac{l}{2}, p_y \pm \frac{l}{2}, \{near, far\}, 1]^T)\},$$



- p_x, p_y = pixel location
- $near$, far are depth values for near and far clipping planes
- C_o = output camera matrix
- l = 3 pixels

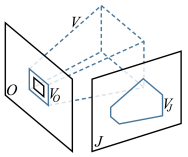
Step 1: Sample Gathering

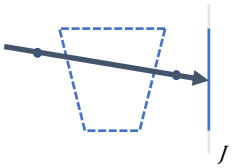


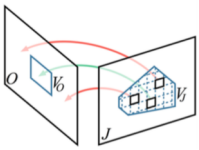
All pixels that project into V_O must reside in V_J.

Step 1: Sample Gathering

- HOWEVER, not all all pixels that reside in V_J project into V_O.
- Why is this?







$$\|p - C_O \cdot C_J^{-1} \cdot [q_x, q_y, q_d, 1]^T\|_1 < \frac{l}{2}$$

- Rasterize all pixels q in V_J
- Check if q projected back into O lies within V_O
- Accept each pixel q that lies within V_O

Step 2: Filtering

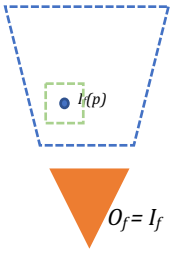
- Some pixels are less trustable – why?

$$\boldsymbol{O}(p) = \Phi(\boldsymbol{S}(p)) = \frac{1}{\boldsymbol{W}} \sum_{s \in \boldsymbol{S}(p)} w(s) s_{rgb}$$

- $w(s)$: application specific weighting function
- W: the sum of all weights

Denoising

- Denoise by averaging multiple observations at same scene point.
- Why not just set $w(s)$ = 1?



Denoising

$$w_{denoise}(s) = \exp\left(-\frac{(s_{ref} - s)^2}{2\sigma^2}\right).$$

- s_{ref}: sample that originates from Input projection into scene space

$$\sigma_{rgb} = 40, \sigma_{xyz} = 10, \sigma_f = 6$$

	r	g	b	x	y	z	f
r	40	...	...	...	...	...	...
g	...	40	...	...	...	...	...
b	...	...	40	...	...	...	...
x	...	...	...	10	...	...	...
y	...	...	...	...	10	...	...
z	...	...	...	...	...	10	...
f	...	...	...	...	...	...	6

Denoising

Super Resolution

- Assumption: each scene point is most clearly recorded when it is observed from as close as possible

$$s_{area} = \|\boldsymbol{C}^{-1} \cdot [p_l, D(p), 1]^T - \boldsymbol{C}^{-1} \cdot [p_r, D(p), 1]^T\|_2^2$$

- p_l and p_r: left and right pixel edge locations
- C: camera matrix
- s_f : sample's frame

$$w_{sr}(s) = \exp\left(-\frac{(s_{ref} - s)^2}{2\sigma^2}\right) \exp\left(-\frac{{s_{area}}^2}{2\sigma_{area}}\right).$$

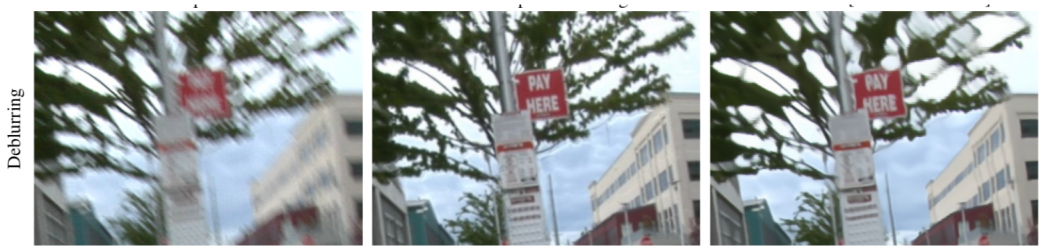
Super Resolution

Deblurring

$$w_{deblur}(s) = \exp\left(-\frac{(s_{ref} - s)^2}{2\sigma^2}\right) \sum_{q \in \boldsymbol{I}^{s_f}} |\nabla \boldsymbol{I}^{s_f}(q)|,$$

- $\nabla I^{s_f}(q)$: gradient operator for the frame that the sample s originated from.
- down-weights contribution from blurry frames
- σ_{rgb} = 200, σ_{xyz} = 10, σ_{f} = 20

Deblurring



Inpainting

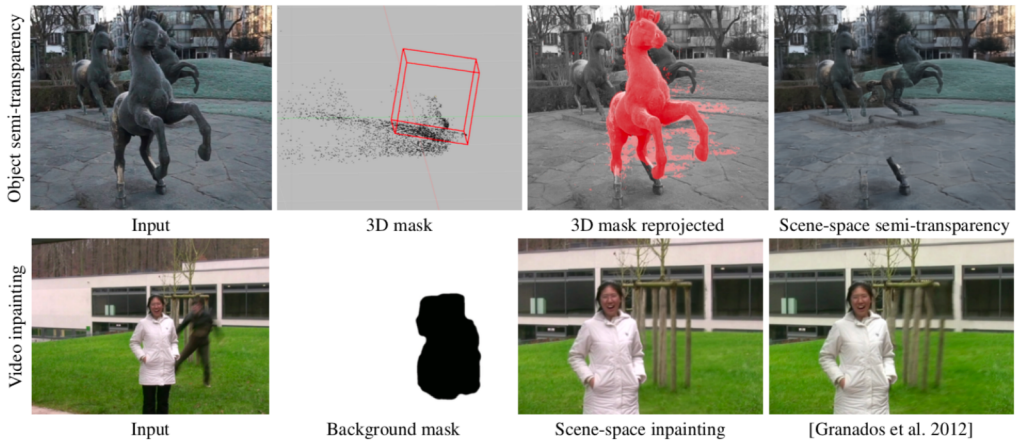
- Requires a user specified mask M where:
  - M(p) = 1 means pixel should be removed.
  - M(p) = 0 otherwise
- Don't have reference s_{ref}.

$$s_{ref} = \frac{1}{|\boldsymbol{S}(p)|} \sum_{s \in \boldsymbol{S(p)}} s$$

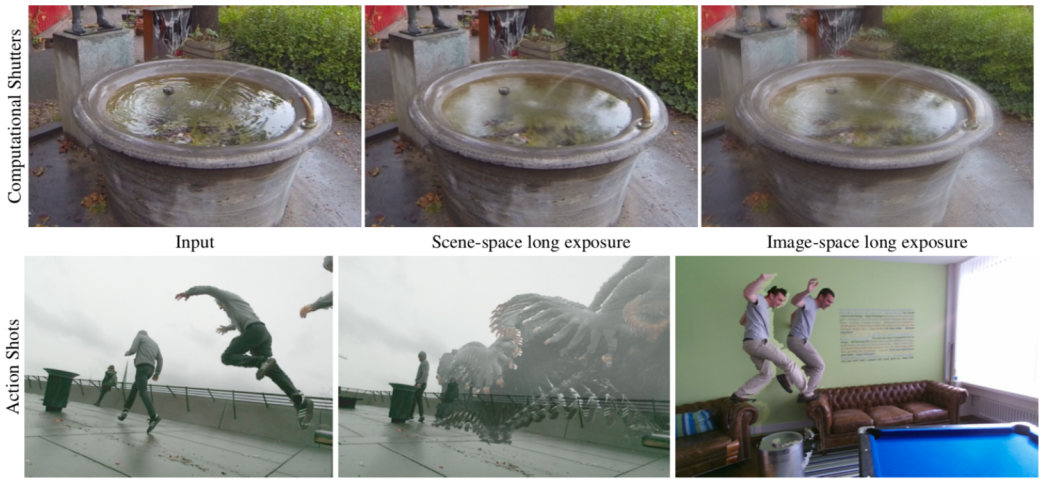
- Weighting function:

$$w_{inpaint}(s) = \begin{cases} \exp\left(-\frac{(s_{ref}-s)^2}{2\sigma^2}\right) & \text{when } \boldsymbol{M}(s_p) = 0 \\ 0 & \text{else} \end{cases}$$

Inpainting



Computational scene-space shutters



Computational scene-space shutters

$$w_{compshutter}(s) = \xi(s_f)$$

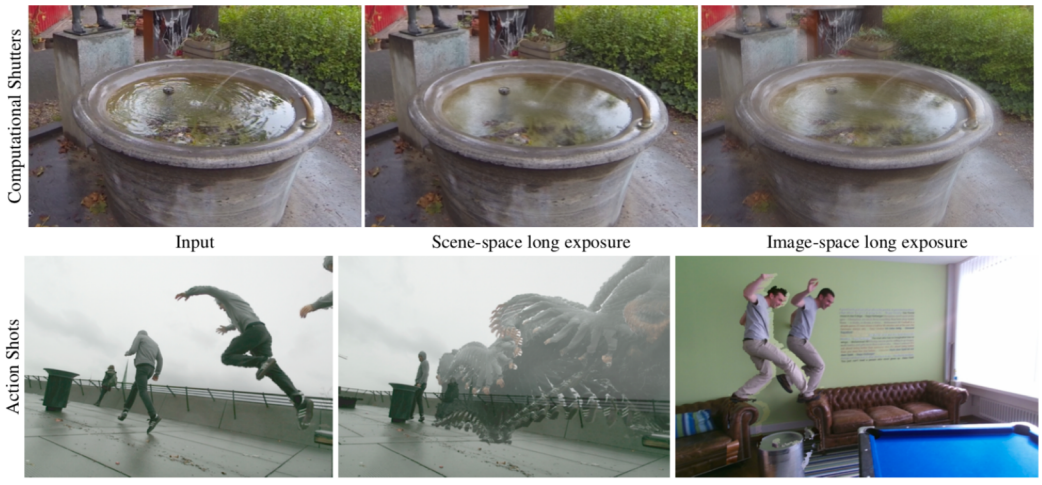
Where ξ(sf): box function in a typical camera

With reasonable depth values:

$$s_{ord} = \#\{q \in \boldsymbol{S}(p) \mid (p_{xyz} - q_{xyz})^2 < (p_{xyz} - s_{xyz})^2\}.$$

$$w_{action} = \xi(s_f) \exp\left(-\frac{s_{ord}^2}{2\sigma_{ord}^2}\right).$$

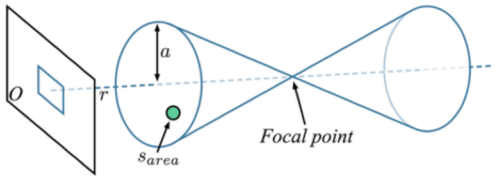
Computational scene-space shutters



Virtual aperture

$a(z) = a_0 + |z_0 - z| * a_s$

- a_0: Thinnest point of cone
- z_0: focal point
- a_s: slope of cone



$$w_{va} = \begin{cases} \frac{s_{area}}{\pi a(r)^2} & \text{when } q < a(r) \\ 0 & \text{else .} \end{cases}$$

Virtual aperture


Sampling Based Scene-Space Video Processing
Felix Klose, Oliver Wang, Jean-Charles Bazin, Marcus Magnor, Alexander Sorkine-Hornung
This video contains narration
© Disney