Multi-view Stereo for 3D Reconstruction

Hosung Son
Vision & Display Systems Lab.
Dept. of Electronic Engineering, Sogang University

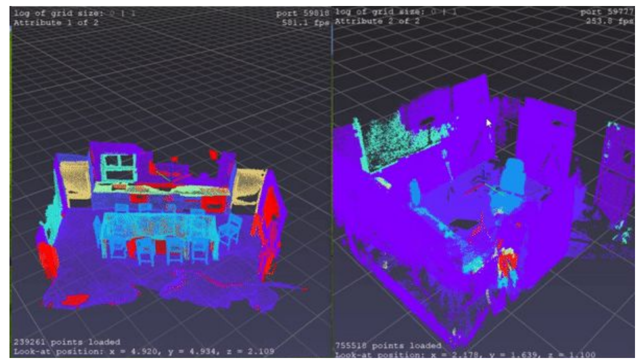
Outline

- Introduction
  - Usage of depth estimation
  - Two-view Stereo & Multi-view Stereo
- Background
  - Stereo Matching
  - Feature Descriptors/Matching Algorithms
  - Camera Parameters
  - Homography
  - Matching Cost Volume
- MVSNet (ECCV2018)
- CVP-MVSNet (CVPR2020 Oral)
- PatchmatchNet (CVPR2021 Oral)

Introduction

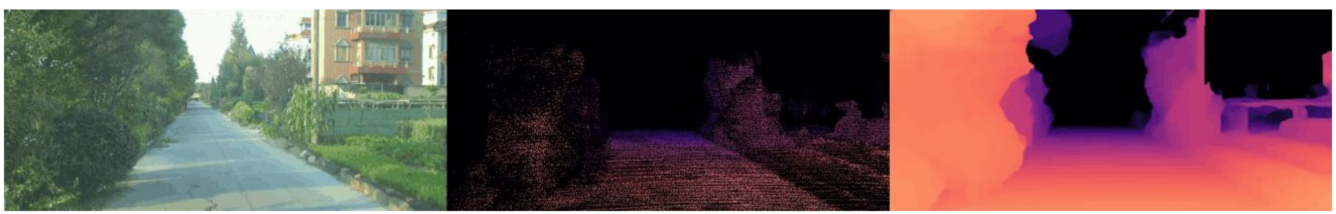
- Usage of depth estimation
  - AR

  - 3D Scene Reconstruction



  - 3D Scanning

  - Autonomous Driving



Introduction

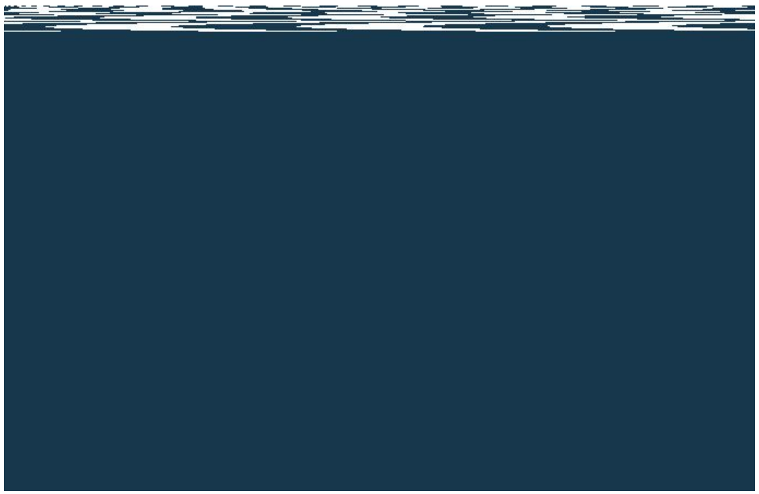
- Two-view Stereo
  - It is similar to Human vision system Fuses a pair of images to get sensation of depth.

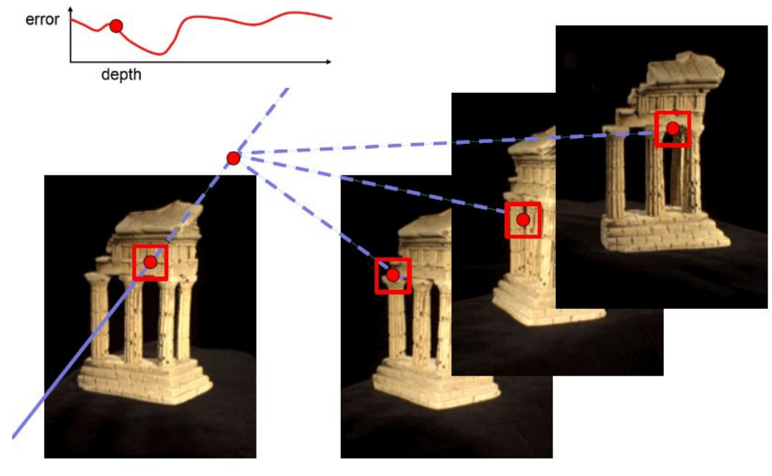
Reconstruction (Reprojection)

Depth cue loss due to many factors such as occlusion from difference viewpoint, reflection, textureless and others.

Introduction

- Multi-view Stereo

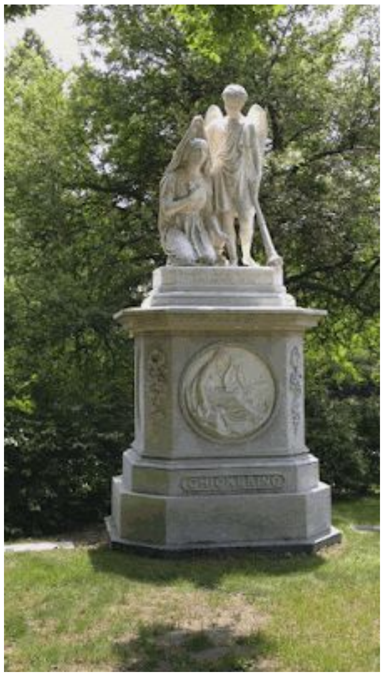




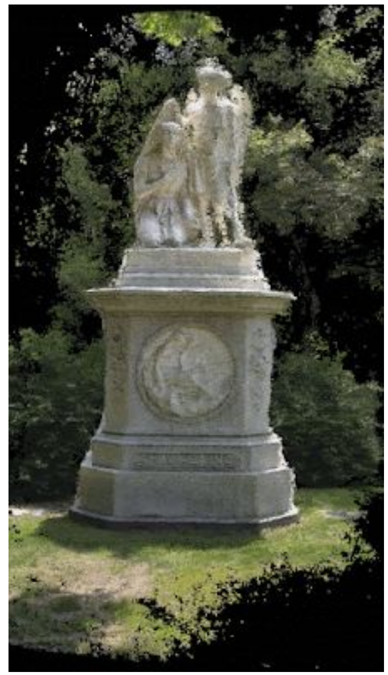
- Interrupted by Occlusion, Non-Lambertian, Reflection, textureless…etc.
- Number of view-points □ Sparse?
- Is deep learning-based model always better than traditional MVS algorithm?

Introduction

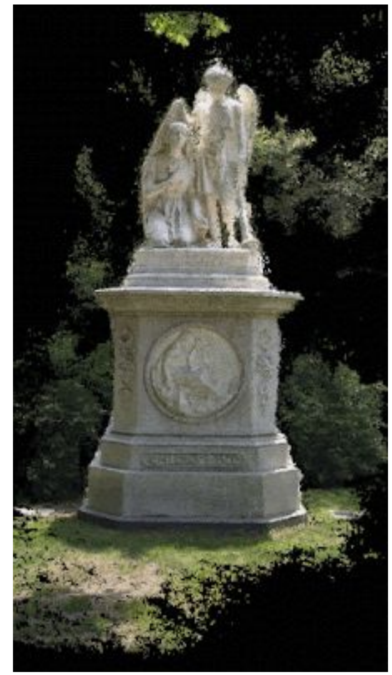
- Quality difference between using 3 or 4 images to reconstruct each 3d points.



Input images



Using 3 images

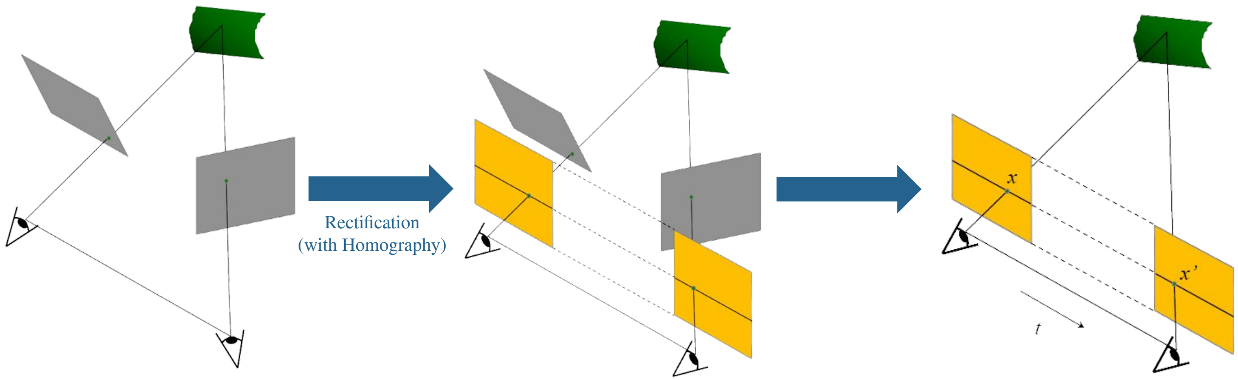


Using 4 images

  - Camera Poses were extracted by Structure-from-Motion(SfM).
  - Dense 3D scene was reconstructed by MVS method.

Background

- Stereo Matching



$$x'^{T}Ex = 0, E = R[t]_{\times}$$

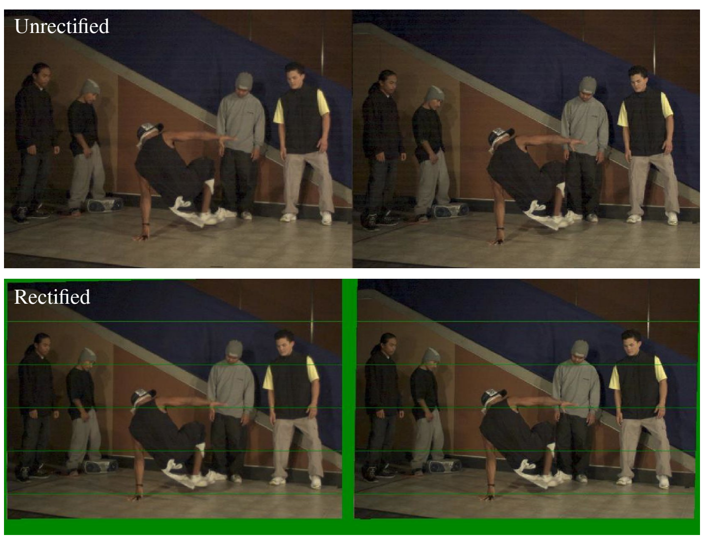
From *Epipolar constraint,*

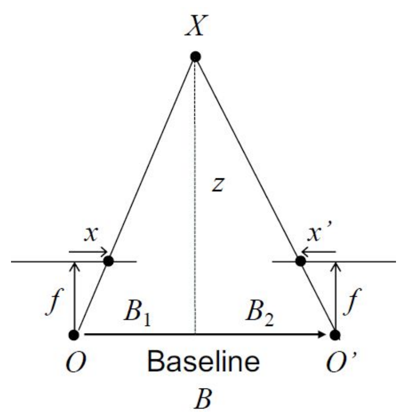
$R{=}I,\ t{=}(T,0,0)$

$$\therefore E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix}$$

$$(u', v', 1)\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = 0$$

$$(u', v', 1)\begin{pmatrix} 0 \\ -T \\ Tv \end{pmatrix} = 0, Tv' = Tv$$





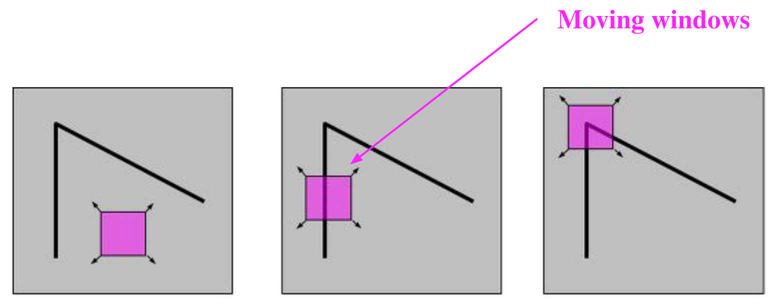
$$\frac{x}{f} = \frac{B_1}{z} \qquad \frac{-x'}{f} = \frac{B_2}{z}$$

$$\frac{x - x'}{f} = \frac{B_1 + B_2}{z}$$

$$\boxed{disparity = x - x' = \frac{B \cdot f}{z}}$$

Background

- Feature(key points) descriptors
  - Harris Corner Detector
    - Traditional Method since 1988.
    - Invariant for translation, illumination and rotation.
    - <u>Variant for scaling.</u>
    - Move fixed size window with 1 px
    - Compute SSD(Sum of Squared Difference) each state and define locally maximum min(E) as “Corner”.



“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

$$E(\Delta x,\Delta y)=\sum_{W}[I(x_i+\Delta x,y_i+\Delta y)-I(x_i,y_i)]^2$$

$$I(x_i+\Delta x,y_i+\Delta y)\approx I(x_i,y_i)+[\ I_x(x_i,y_i)\ I_y(x_i,y_i)\ ]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

$$E(\Delta x,\Delta y)=\sum_{W}[I(x_i+\Delta x,y_i+\Delta y)-I(x_i,y_i)]^2$$

$$\approx\sum_{W}[I(x_i,y_i)+[\ I_x(x_i,y_i)\ I_y(x_i,y_i)\ ]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}-I(x_i,y_i)]^2$$

$$=[\ \Delta x\ \Delta y\ ]\begin{bmatrix}\sum_{W}I_x(x_i,y_i)^2 & \sum_{W}I_x(x_i,y_i)I_y(x_i,y_i)\\ \sum_{W}I_x(x_i,y_i)I_y(x_i,y_i) & \sum_{W}I_y(x_i,y_i)^2\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

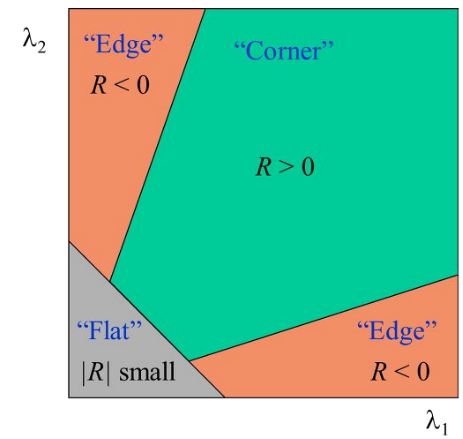
$$=[\ \Delta x\ \Delta y\ ]M\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

$$R=det(M)-k*tr(M)^2$$

$$det(M)=\lambda_1\lambda_2$$

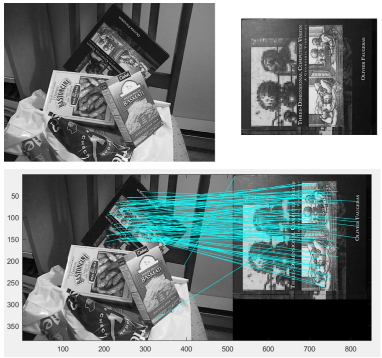
$$tr(M)=\lambda_1+\lambda_2$$

$$k\in[0.04,0.06]$$

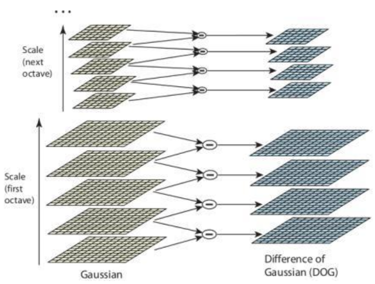


Background

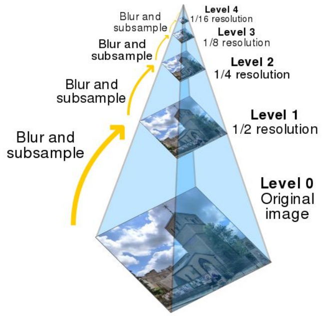
- Feature(key points) Descriptors
  - SIFT (Scale-Invariant Feature Transform)
    - Invariant for rotation, illumination, translation, scaling.
    - Algorithm
      1. Make Scale Space
         ① 4 stages : 2x, original, ½, ¼ size images
         ② Generally Blurring them with Gaussian Filtering
      2. Computing DoG(Difference of Gaussian) with 3 images in each stage
         $$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial^2 \mathbf{x}^2}\mathbf{x}$$
      3. Finding key points
      4. Remove bad key points
         $R = tr(\mathbf{H})^2/\det(\mathbf{H})$,
         If $R > \frac{(r_{th}+1)^2}{r_{th}}$, It is Poor. ($r_{th}$=10)
  - SURF (Speed-Up Robust Features)
    - Faster, more robust method.



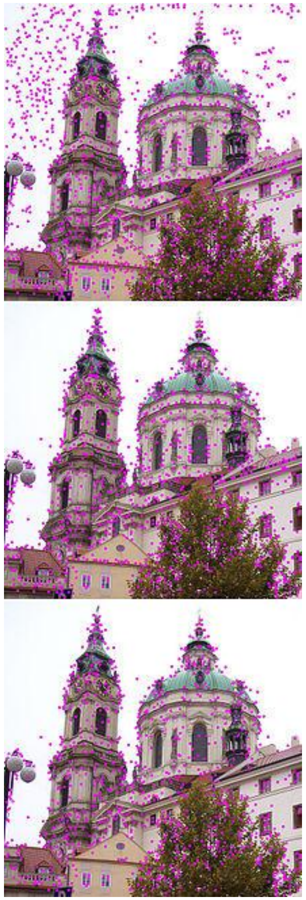
Matching example with SIFT



DoG Computation



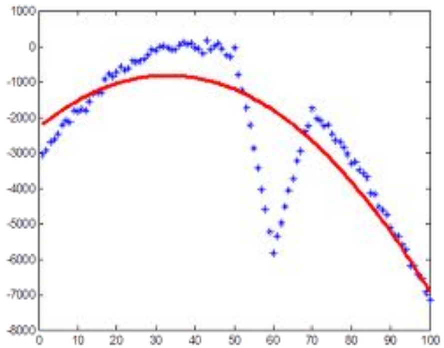
Scale Space



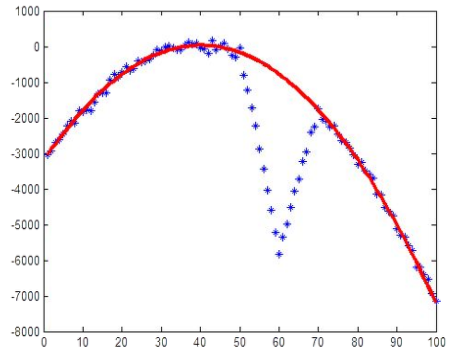
Removing bad key points

Background

- Feature(key points) Matching
  - Brute Force Matcher
    - In BF Matcher, we have to match descriptor of all features in an image to descriptors of all features in another image.
    - It is extremely expensive, however, doesn’t guarantee getting an optimal solution.
  - RANSAC (RANdom Sample Consensus)
    - Randomly choose some samples and make a model with them.
    - Compute distance and count the number of samples which loss is lower than threshold.
    - Select a model which has the maximum number of consensus iteratively.



L2 quadrature Regression



RANSAC

서강대학교
SOGANG UNIVERSITY

VDS LAB

Background

- Camera Parameters
  - Intrinsic

    $$- K = \begin{bmatrix} f_x & \gamma & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \leftarrow \text{Camera Calibration}$$

  - Extrinsic camera parameters

    $$- \begin{bmatrix} R_{3\times3} & t_{3\times1} \\ O_{1\times3} & 1 \end{bmatrix}_{4\times4}$$

    - $R_{3\times3}$: Rotation matrix

    - $t_{3\times1}$: Translation vector

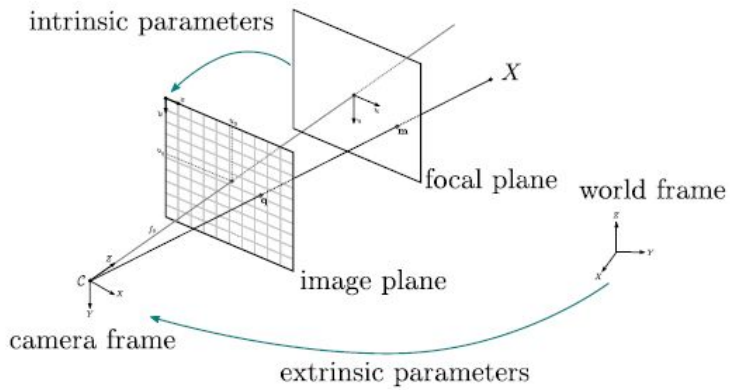
  - Essential Matrix (E)

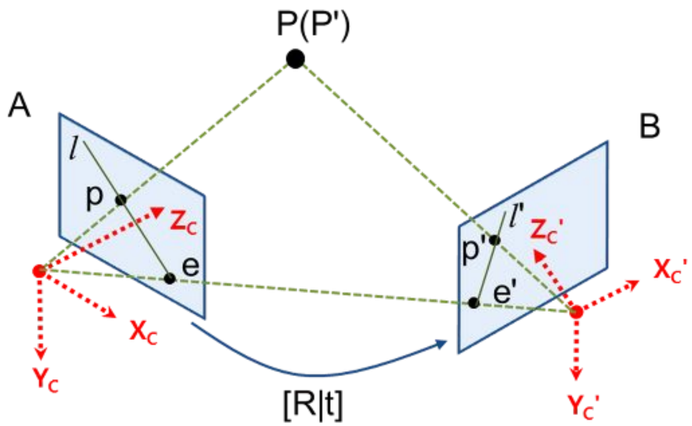
    $E = R[t]_\times$

  - Fundamental Matrix (F)

    8-point Algorithm

    $F = (K^T)^{-1} E K^{-1}$





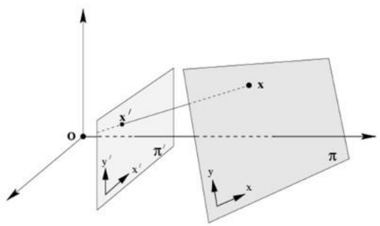
Epipolar Geometry

Background

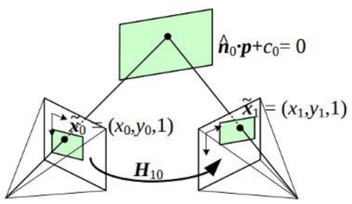
- Homography

$$\mathbf{H}_i(d) = \mathbf{K}_i \cdot \mathbf{R}_i \cdot \left(\mathbf{I} - \frac{(\mathbf{t}_1 - \mathbf{t}_i)\cdot \mathbf{n}_1^T}{d}\right)\cdot \mathbf{R}_1^T \cdot \mathbf{K}_1^T \qquad s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

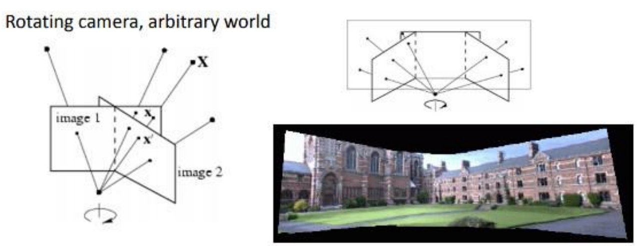
  - Homography matrix is a relation between two planar surface in space.
  - There are diverse practical applications such as image rectification, registration.



planar surface and image plane



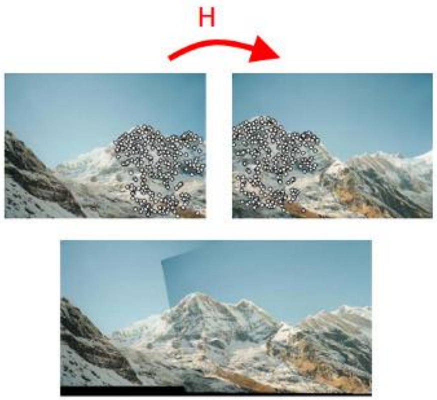
viewed by two camera positions



rotating camera and image stitching

  - H is 3 by 3 matrix with 8 DoF since it is generally normalized.

$$h_{33} = 1 \ or \ h_{11}^2 + h_{12}^2 + h_{13}^2 + h_{21}^2 + h_{22}^2 + h_{23}^2 + h_{31}^2 + h_{32}^2 + h_{33}^2 = 1$$



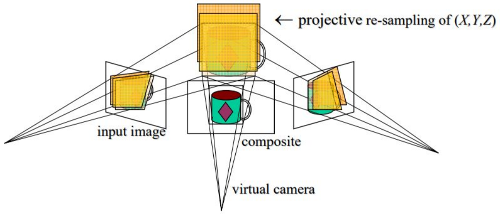
Background

- Matching Cost Volume
  - Plane-sweep algorithm (aggregation function : ZNCC)
    - Map each target image I_k to the reference image I_{ref} for each depth plane Π_m with the homography $H^{-1}_{\Pi_m,P_k}$ giving the warped image $\check{I}_{k,m}$
    - Compute the <u>similarity</u> between I_{ref} and each $\check{I}_{k,m}$ using Zero-mean Normalized Cross Correlation(ZNCC) between small patches W around each pixel.
    - Compute the figure-of-merit for each depth plane by combining the similarity measurements for each image k

$$M(\mathrm{u},\mathrm{v},\Pi_m) = \sum_k ZNCC(I_{ref}, \check{I}_{k,m})$$

    - For each pixel, select the <u>depth plane</u> with best fit

$$\tilde{\Pi}(u,v) = argmax\, M(u,v,\Pi_m)$$

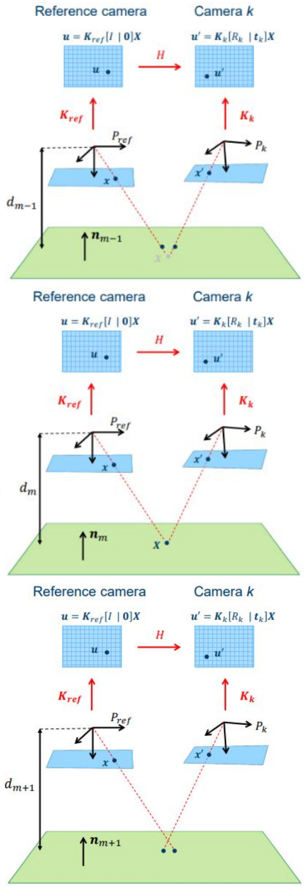


Composite homography after Plane-sweep

$$ZNCC = \frac{\sum_{i=1}^{MN}(f_i-\bar{f})(g_i-\bar{g})}{\sqrt{\sum_{i=1}^{MN}(f_i-\bar{f})^2 \sum_{i=1}^{MN}(g_i-\bar{g})^2}}$$

f_i: i^{th} pixel intensity of I_{ref}, $\bar{f}$: mean of I_{ref} intensity

g_i: i^{th} pixel intensity of $\check{I}_{k,m}$, $\bar{f}$: mean of $\check{I}_{k,m}$ intensity

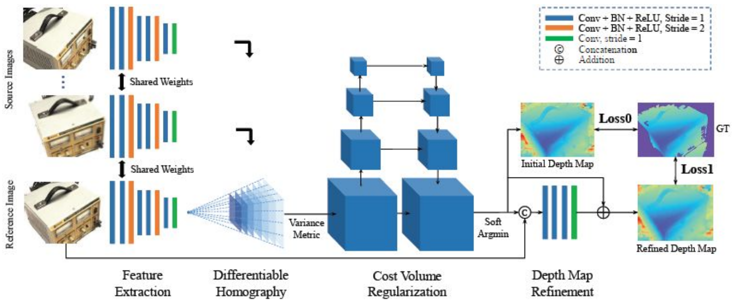


Visualization of plane sweeping

Comparison of Various Stereo Vision Cost Aggregation Methods : https://www.ijeit.com/vol%202/Issue%208/IJEIT1412201302_45.pdf

MVSNet (ECCV2018)

- Standard Multi-view Stereo Network for 3D reconstruction



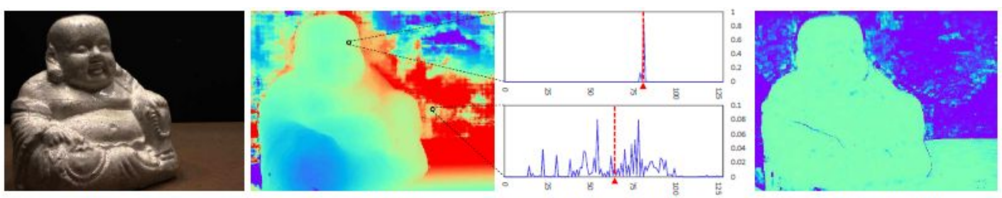
MVSNet Architecture

$$\mathbf{H}_i(d) = \mathbf{K}_i \cdot \mathbf{R}_i \cdot \left(\mathbf{I} - \frac{(\mathbf{t}_1 - \mathbf{t}_i) \cdot \mathbf{n}_1^T}{d}\right) \cdot \mathbf{R}_1^T \cdot \mathbf{K}_1^T$$

$$\mathbf{C} = \mathcal{M}(\mathbf{V}_1, \cdots, \mathbf{V}_N) = \frac{\sum_{i=1}^{N} (\mathbf{V}_i - \overline{\mathbf{V}_i})^2}{N}$$

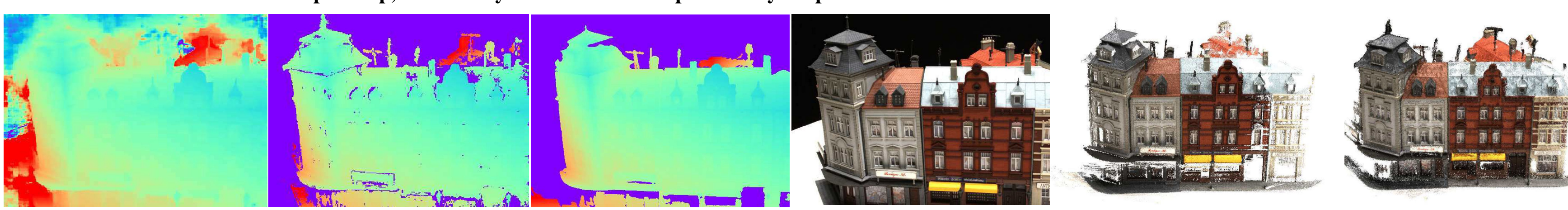
$$\mathbf{D} = \sum_{d=d_{min}}^{d_{max}} d \times \mathbf{P}(d)$$

$$Loss = \sum_{p \in \mathbf{p}_{valid}} \underbrace{\|d(p) - \hat{d}_i(p)\|_1}_{Loss0} + \lambda \cdot \underbrace{\|d(p) - \hat{d}_r(p)\|_1}_{Loss1}$$



(a) Reference image (b) Inferred depth map (c) Probability distribution (d) Probability Map

Illustration of inferred depth map, Probability distributions and probability Map



Inferred depth map | Filtered depth map | GT depth map | Reference image | Fused point cloud | GT point cloud

Results of MVSNet for DTU Dataset

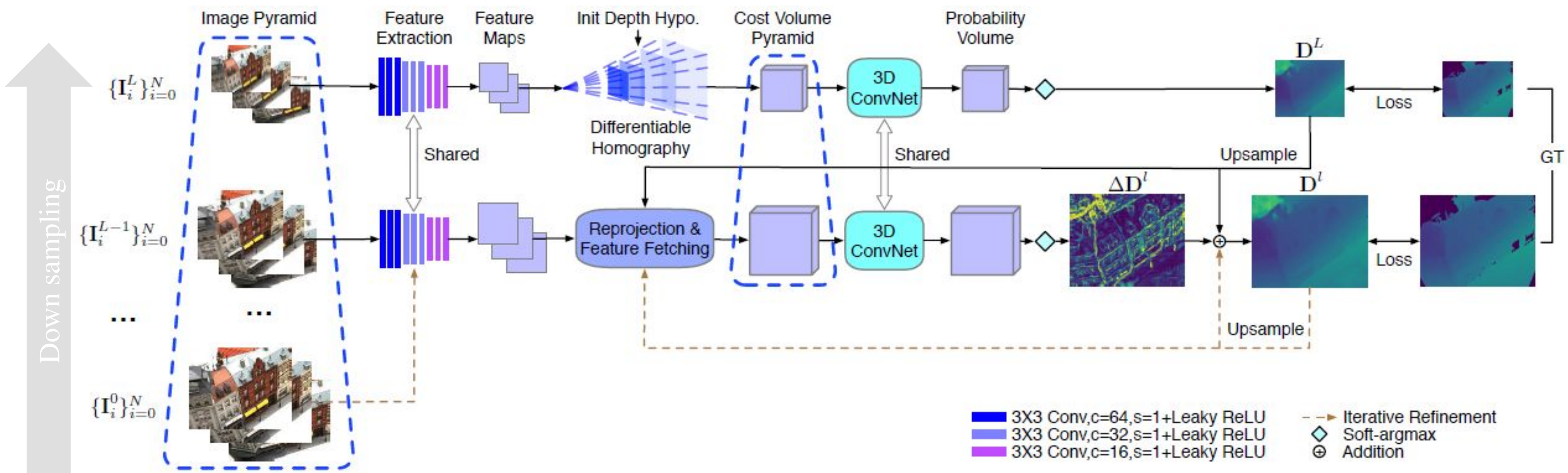
MVSNet (ECCV2018)

- Matching cost volume with Feature map variance
  - $C = M(V_1, \cdots, V_N) = \frac{1}{N}\sum_{i=1}^{N}(V_i - \overline{V_i})^2$, V_i: i^{th} *warped feature volumes*
  - Setting Mean of Squared Difference to Cost volume
- Depth Refinement
  - Depth residual learning at the end of Network.
  - Use reference image as a <u>guidance</u> to refine the initial depth map.
  - Concatenate the <u>initial depth map</u> and <u>resized reference image</u> as a 4-channel input

    → Pass to the <u>three</u> 32-channel 2D CNN layers to learn the depth residual.
  - Add the residual and initial depth map, finally generate refined depth map.
- Limitations
  - Computational Cost (GPU Memory demand : around 11GB)
  - Time consuming (takes 230s for one scan → 4.7s per view)
  - Occlusion

CVP-MVSNet (CVPR2020 Oral)

- Network Structure



- Contribution
  - Computational efficient depth inference network for MVS.
  - Cost volume pyramid in a coarse-to-fine manner.
  - 6x-faster than current SOTA & better accuracy.

CVP-MVSNet (CVPR2020 Oral)

- Cost Volume Pyramid
  - Common MVS methods generate a Cost volume with fixed resolution to inference depth map.
  - Instead, proposed method generate multi-level Cost volume pyramid.
  - Estimate depth for each cost volumes and refine them using residual depth iteratively.
  - The resolution of depth map increases generally.

    → High resolution depth map with high accuracy.
  - Cost volume equation at level L

$$\mathbf{C}_d^L = \frac{1}{(N+1)} \sum_{i=0}^{N} (\tilde{\mathbf{f}}_{i,d}^L - \bar{\mathbf{f}}_d^L)^2$$

  - Similar to MVSNet, 3D convolution was applied to the constructed cost volume pyramid and output Depth Probability Map $P(d)$ from $softmax$ operation. (d: sampled depth plane)

CVP-MVSNet (CVPR2020 Oral)

- Depth Map Inference
  - Coarse Depth map

$$\mathbf{D}^L(\mathbf{p}) = \sum_{m=0}^{M-1} d\mathbf{P}^L_{\mathbf{p}}(d)$$

- L^{th} level Depth estimate for each pixel **p**
- $d = d_{min} + m(d_{max} - d_{min})/M$: sampled depth

  - Refined Depth map

$$\mathbf{D}^l(\mathbf{p}) = \mathbf{D}^{l+1}_{\uparrow}(\mathbf{p}) + \sum_{m=-M/2}^{(M-2)/2} r_{\mathbf{p}} \mathbf{P}^l_{\mathbf{p}}(r_{\mathbf{p}})$$

- $l \in \{L-1, L-2, \cdots, 0\}$
- $r_{\boldsymbol{p}} = m \cdot \Delta d^l_{\boldsymbol{p}}$: depth residual hypothesis ($m \in \{0,1,2,\cdots,M-1\}, M = 48\ at\ experiment$)
- No depth map refinement after proposed pyramidal depth estimation can obtain good results.

  - Loss Function

$$Loss = \sum_{l=0}^{L} \sum_{\mathbf{p} \in \Omega} \|\mathbf{D}^l_{GT}(\mathbf{p}) - \mathbf{D}^l(\mathbf{p})\|_1$$

CVP-MVSNet (CVPR2020 Oral)

- Performance

  - Comparison of reconstruction quality

Method	Input Size	Depth Map Size	Acc.(mm)	Comp.(mm)	Overall(mm)	*f-score*(0.5mm)	GPU Mem(MB)	Runtime(s)
Point-MVSNet[5]	1280x960	640x480	0.361	0.421	0.391	84.27	8989	2.03
Ours-640	640x480	640x480	0.372	0.434	0.403	82.44	**1416**	**0.37**
Point-MVSNet[5]	1600x1152	800x576	0.342	0.411	0.376	-	13081	3.04
Ours-800	800x576	800x576	0.340	0.418	0.379	86.82	**2207**	**0.49**
MVSNet[42]	1600x1152	400x288	0.396	0.527	0.462	78.10	22511	2.76
R-MVSNet[43]	1600x1152	400x288	0.383	0.452	0.417	83.96	**6915**	5.09
Point-MVSNet[5]	1600x1152	800x576	0.342	0.411	0.376	-	13081	3.04
Ours	1600x1152	1600x1152	**0.296**	**0.406**	**0.351**	**88.61**	8795	**1.72**

  - Metric

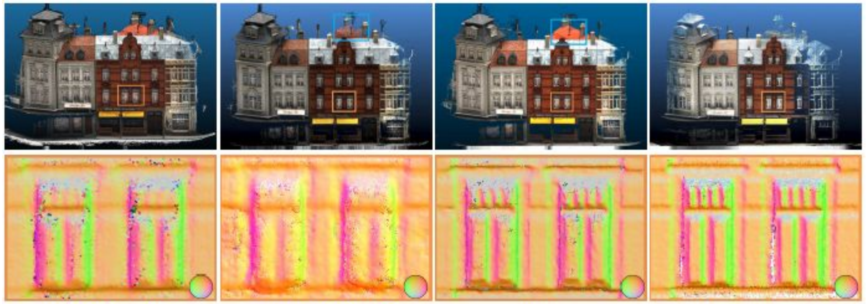
$f - score$:

$$e_{r \to \mathcal{G}} = \min_{g \in \mathcal{G}} |\boldsymbol{r} - \boldsymbol{g}|, \; e_{g \to \mathcal{R}} = \min_{r \in \mathcal{R}} |\boldsymbol{g} - \boldsymbol{r}|$$

$$P(d) = \frac{100}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} [e_{r \to \mathcal{G}} < thrs.]$$

$$\mathrm{R}(d) = \frac{100}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} [e_{g \to \mathcal{R}} < thrs.]$$

$$F(d) = \frac{2P(d)R(d)}{P(d)+R(d)}$$


R-MVSNet [43] Point-MVSNet [5] Ours Ground truth

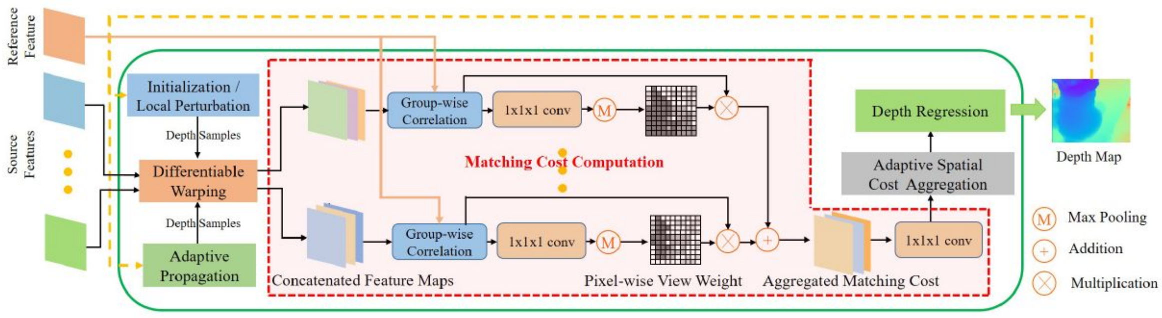
Qualitative results of scan 9 of DTU dataset.

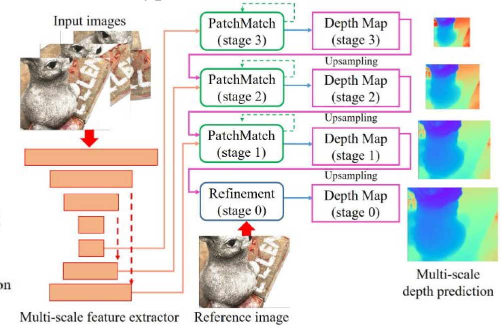
Accuracy: Distance from estimated point clouds to the ground truth ones.

Completeness: Distance from ground truth point clouds to the estimated ones.

PatchmatchNet (CVPR2021 Oral)

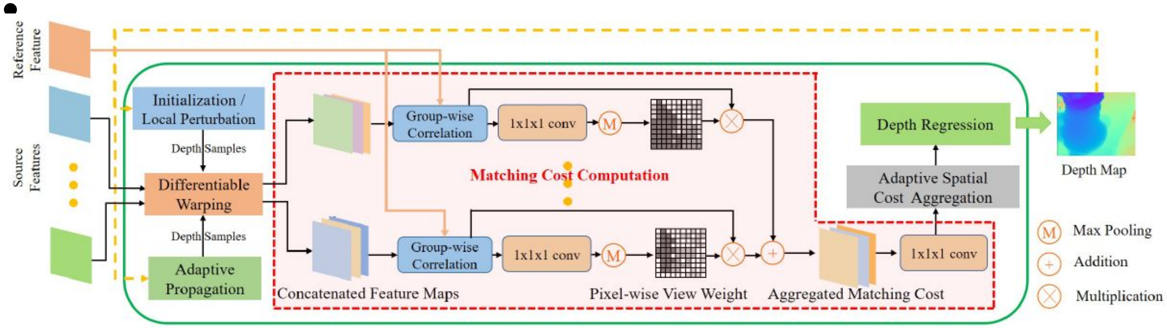
At only Stage 3, random depth hypotheses in *initialization* are used.





- Network Structure
  - Refrain from parameterizing the per-pixel hypothesis as a slanted plane.
  - Instead, adaptive evaluation was used to organize the spatial pattern within the window over which matching costs are computed.
- 3-steps in Learning-based Patchmatch
  1. Initialization and Local Perturbation: generate random hypotheses.
  2. Adaptive propagation: propagate hypotheses to neighbors.
  3. Adaptive evaluation: compute the matching costs for all hypotheses, choose best solutions.

PatchmatchNet (CVPR2021 Oral)



Implementation Details

- Image resolution : 640×512
- \# of input images : N=5
- Iteration # of Patchmatch on stage 3, 2, 1 : 2, 2, 1
- $D_f = 48$
- $R_3 = 0.38, R_2 = 0.09, R_1 = 0.04$
- $N_3 = 16$, $N_2 = N_1 = 8$
- K_p on stage 3, 2, 1 : 16, 8, 0
- $K_e = 9$

- Initialization
  - First, sample per pixel D_f depth hypotheses in the *inverse* depth range.

    →It helps model be applicable to complex and large-scale scenes.
  - Divide the range into D_f intervals and ensure that each interval is covered by one hypothesis.
- Local Perturbation
  - Generating per pixel N_k hypotheses uniformly in the normalized inverse depth range R_k.
  - Decrease gradually R_k for finer stage.
  - Sampling around the previous estimation can refine result locally and correct wrong estimates.

서강대학교 SOGANG UNIVERSITY

VDS LAB

PatchmatchNet (CVPR2021 Oral)

- Adaptive Propagation
  - Based on Deformable Convolution Networks.
    - DCN : offset based flexibly sampling along with Convolution Networks.
  - To gather K_p depth hypothesis for pixel $\mathbf{p}$ in the reference image, model learns additional 2D offsets $\{\Delta\mathbf{o}_i(\mathbf{p})\}_{i=1}^{K_p}$ that are applied on top of fixed 2D offsets $\{\mathbf{o}_i\}_{i=1}^{K_p}$ organized as a grid.
  - Apply a 2D CNN on the reference feature map $\mathbf{F}_0$ to learn additional 2D offsets for each pixel $\mathbf{p}$.
  - Depth hypotheses $\mathbf{D}_p(\mathbf{p})$ via bilinear interpolation:

$$\mathrm{D}_p(\mathrm{p}) = \{\mathrm{D}(\mathrm{p} + \mathrm{o}_i + \Delta\mathrm{o}_i(\mathrm{p}))\}_{i=1}^{K_p}$$

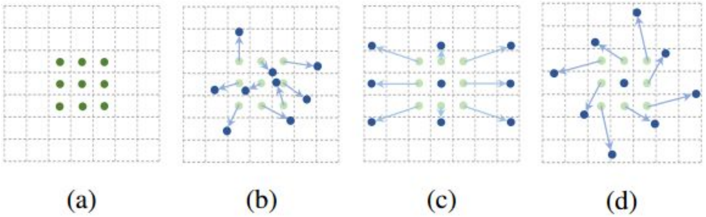
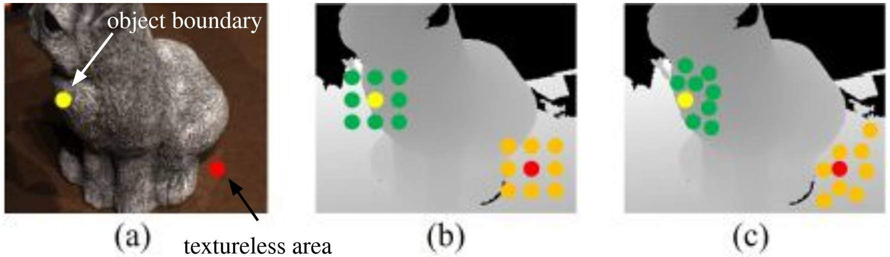


Illustration of the sampling locations in 3x3 standard and deformable convolutions.



(a) reference image
(b) Fixed sampling locations
(c) Adaptive sampling locations

PatchmatchNet (CVPR2021 Oral)

- Adaptive Evaluation
  - Differentiable Warping
  - Matching cost computation
    - Group-wise correlation
      1. $\mathbf{F}_0(\mathbf{p}), \mathbf{F}_i(\mathbf{p}_{i,j})$: features in the ref, src feature maps. (view i, j – th set of depth hypothesis)
      2. divide feature maps' feature channels evenly into G groups.
      3. $< \cdot , \cdot >$: inner product
      4. $\mathrm{S}_i(\mathbf{p}, j)^g = \frac{G}{C}\langle \mathbf{F}_0(\mathbf{p})^g, \mathbf{F}_i(\mathbf{p}_{i,j})^g \rangle \in \mathbb{R}^G$: g^{th} group similarity (C: # of channel)
    - Pixel-wise view weight network
      1. Composed of 3D convolution layers with 1x1x1 kernels and sigmoid.
      2. Takes the initial set of similarities S_i to output a number between 0 and 1 per pixel and depth hypothesis.
      3. $w_i(\mathbf{p}) = \max\{\mathbf{P}_i(\mathbf{p}, j) | j = 0, 1, \cdots, D-1\}$: view weights for pixel $\mathbf{p}$ and source image I_i
      4. $\bar{\mathrm{S}}(\mathbf{p}, j) = \frac{\sum_{i=1}^{N-1} w_i(\mathbf{p}) \cdot \mathrm{S}_i(\mathbf{p}, j)}{\sum_{i=1}^{N-1} w_i(\mathbf{p})}$: final per group similarities for pixel $\mathbf{p}$ and j^{th} hypothesis
      5. Finally, compose $\bar{\mathrm{S}}(\mathbf{p}, j)$ for all pixels and hypothesis into $\bar{\mathrm{S}} \in \mathbb{R}^{W \times H \times D \times G}$
      6. Apply a small network with 3D convolution and 1x1x1 kernels to obtain single cost $\mathbf{C} \in \mathbb{R}^{W \times H \times D}$

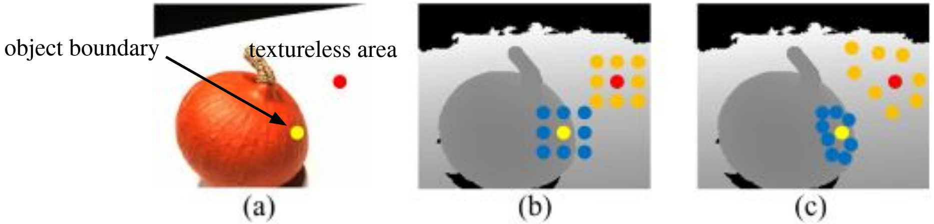
PatchmatchNet (CVPR2021 Oral)

- Adaptive Evaluation
  - Adaptive spatial cost aggregation
    - K_e: spatial window
    - w_k: feature weight at a pixel **p** based on the feature similarity.
    - d_k: depth weight based on the similarity of depth hypotheses.
    - Aggregated spatial cost:

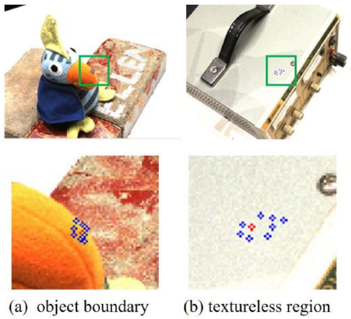
$$\tilde{\mathbf{C}}(\mathbf{p}, j) = \frac{1}{\sum_{k=1}^{K_e} w_k d_k} \sum_{k=1}^{K_e} w_k d_k \mathbf{C}(\mathbf{p}+\mathbf{p}_k+\Delta\mathbf{p}_k, j)$$

  - Depth regression
    - Apply $softmax$ to (negative) cost $\tilde{\mathbf{C}}$ to generate a probability **P**.
    - Regressed depth value at pixel **p**:

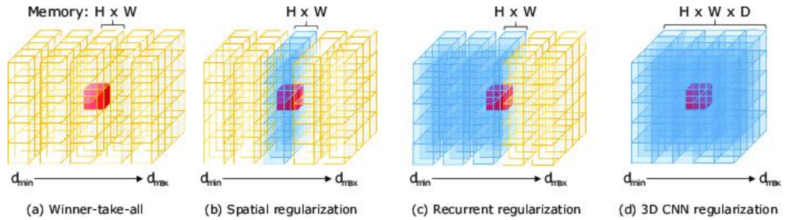
$$\mathbf{D}(\mathbf{p}) = \sum_{j=0}^{D-1} d_j \cdot \mathbf{P}(\mathbf{p}, j)$$



(a) reference image
(b) Fixed sampling locations
(c) Adaptive sampling locations



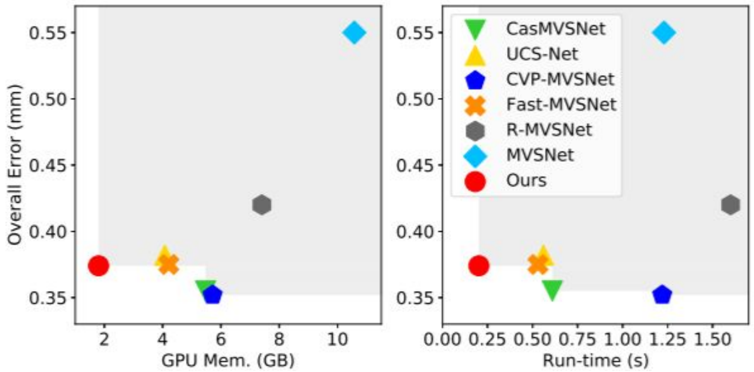
Visualization of adaptive propagation of two typical situations



Cost volume regularization schemes
(d) Captures context in all dimensions by using 3D CNN

PatchmatchNet (CVPR2021 Oral)

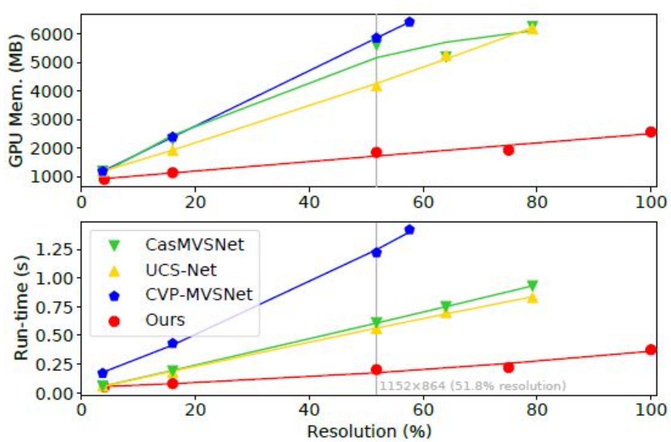
- Performance



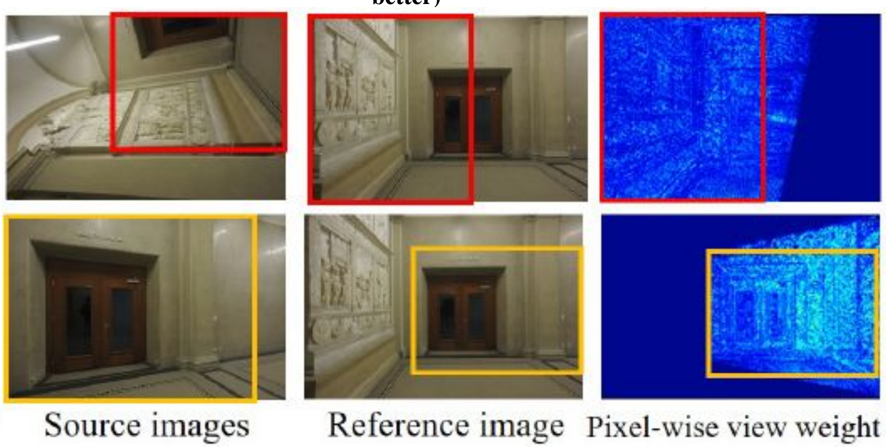
Comparison with SOTA learning based MVS methods

Methods	Acc.(mm)	Comp.(mm)	Overall(mm)
Camp [4]	0.835	0.554	0.695
Furu [15]	0.613	0.941	0.777
Tola [35]	0.342	1.190	0.766
Gipuma [16]	**0.283**	0.873	0.578
SurfaceNet [20]	0.450	1.040	0.745
MVSNet [42]	0.396	0.527	0.462
R-MVSNet [43]	0.383	0.452	0.417
CIDER [39]	0.417	0.437	0.427
P-MVSNet [28]	0.406	0.434	0.420
Point-MVSNet [6]	0.342	0.411	0.376
Fast-MVSNet [44]	0.336	0.403	0.370
CasMVSNet [17]	0.325	0.385	0.355
UCS-Net [7]	0.338	0.349	**0.344**
CVP-MVSNet [41]	0.296	0.406	0.351
Ours	0.427	**0.277**	0.352

Results on DTU's evaluation set (lower is better)



Relating GPU memory and run-time to the input resolution



Visualization of pixel-wise view weight on a scene from ETH3D