Accurate, Dense, and Robust Multi-View Stereopsis

Authors: Yasutaka Furukawa, Jean Ponce
Presenter: Nail Ibrahimli
In one slide :) 

1. Detect keypoints
2. Triangulate a sparse set of initial matches
3. Iteratively expand matches to nearby locations
4. Use visibility constraints to filter out false matches
5. Perform surface reconstruction
Patch Model

c(p): center of the patch
n(p): normal of the patch
R(p): reference image with p

\[
\begin{align*}
c(p) & \leftarrow \{\text{Triangulation from } f \text{ and } f'\}, \\
n(p) & \leftarrow \frac{c(p)O(I_i)}{|c(p)O(I_i)|}, \\
R(p) & \leftarrow I_i.
\end{align*}
\]
normalized cross correlation: quick glance

\[ \hat{f} = \frac{f - \bar{f}}{\sqrt{\sum (f - \bar{f})^2}} \]

\[ \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum (g - \bar{g})^2}} \]

\[ \text{NCC}(f,g) = C_{fg}(\hat{f}, \hat{g}) = \sum_{[i,j] \in R} \hat{f}(i,j)\hat{g}(i,j) \]
Photometric Discrepancy Function

\[ h(p, I, R(p)) = 1 - NCC(p, I, R(p)) \]

\[ g(p) = \frac{1}{|V(p) \setminus R(p)|} \sum_{I \in V(p) \setminus R(p)} h(p, I, R(p)), \]

V(p): initial set of images where patch p is potentially visible
Photometric Discrepancy Function

\[ V^*(p) = \{ I | I \in V(p), h(p, I, R(p)) \leq \alpha \}, \]
\[ g^*(p) = \frac{1}{|V^*(p) \setminus R(p)|} \sum_{I \in V^*(p) \setminus R(p)} h(p, I, R(p)). \]

\( V^*(p) \): set of images where patch is truly visible
Patch optimization

\[ h(p, I, R(p)) = 1 - NCC(p, I, R(p)) \]

\[ g^*(p) = \frac{1}{|V^*(p) \setminus R(p)|} \sum_{I \in V^*(p) \setminus R(p)} h(p, I, R(p)) \]

Optimize over \( c(p) \) and \( n(p) \) that minimizes \( g^*(p) \)
**Image model**

V(p): set of images where patch may be visible

V*(p): set of images where patch truly visible

C_i(x,y): regular grid cell $\beta \times \beta$ pixels

Q_i(x,y): the set of "may be" visible patches that projects to C_i(x,y)

Q*_i(x,y): the set of "truly visible" patches that projects to C_i(x,y)
Flow diagram

Initialization, Feature Detection & Matching → Expansion → Filtering → Surface Reconstruction

Patch Model
Quick glance to corners: Aperture problem
Feature Detection

1. Divide grid to cells (32x32)

2. Use Harris Detector and DoG to find corners

3. Try to find 4 good corners in each cell (uniform coverage)
Quick glance: typical feature matching pipeline

Detection → Description → Matching → Filtering

\[ p'^T K'^-T [T_x] R K^{-1} p = 0 \]
Feature Matching

1. Epipolar line test for right matches
2. Initialization of patches

\[
\begin{align*}
c(p) & \leftarrow \{\text{Triangulation from } f \text{ and } f'\}, \\
n(p) & \leftarrow \frac{c(p)O(I_i)}{|c(p)O(I_i)|}, \\
R(p) & \leftarrow I_i. \\
V(p) & \leftarrow \left\{I | n(p) \cdot c(p)O(I)/|c(p)O(I)| > \cos(\alpha)\right\} \\
V^*(p) & = \left\{I | I \in V(p), h(p,I,R(p)) \leq \alpha\right\}
\end{align*}
\]

3. Refine patch geometry
Feature Matching

4. Update the $V(p)$ and $V^*(p)$ with refined patch geometry

5. Check if patch truly visible in at least $\gamma$ images

6. Add valid patches to corresponding $Q$ and $Q^*$
Expansion

1. Identify neighboring cells for possible expansion
   \[ C(p) = \{ C_i(x', y') | p \in Q_i(x, y), |x - x'| + |y - y'| = 1 \} \]

2. Test if there is already patch very close to that region
   \[ |(c(p) - c(p')) \cdot n(p) + (c(p) - c(p')) \cdot n(p')| < 2\rho \]

3. Test for depth discontinuity
Expansion

4. Initialize candidate patch
5. Refine patch geometry
6. Update the V(p) and V*(p) with refined patch geometry (loosen thresholds)
7. Check if patch truly visible in at least $\gamma$ images
8. Add valid patches to corresponding Q and Q*

| Input: | Patches $P$ from the feature matching step. |
| Output: | Expanded set of reconstructed patches. |

While $P$ is not empty
Pick and remove a patch $p$ from $P$;
For each image cell $C_i(x,y)$ containing $p$
Collect a set $C$ of image cells for expansion;
For each cell $C_i(x',y')$ in $C$
   // Create a new patch candidate $p'$
   $n(p') \leftarrow n(p), \; R(p') \leftarrow R(p), \; V(p') \leftarrow V^*(p');$
   Update $V^*(p')$; // Eq. (2)
   Refine $c(p')$ and $n(p')$; // (Sect.II-C)
   Add visible images (a depth-map test) to $V(p')$;
   Update $V^*(p')$; // Eq. (2)
   If $|V^*(p')| < \gamma$
      Go back to For-loop (failure);
   Add $p'$ to $P$;
   Add $p'$ to corresponding $Q_f(x,y)$ and $Q_f^*(x,y)$;
Filtering

First filter: Global visibility consistency

$$|V^*(p)| (1 - g^*(p)) < \sum_{p_i \in U(p)} 1 - g^*(p_i).$$

Second filter: Depth map test
check if patch truly visible in at least $\gamma$ images after depth map test

Third filter: Check if patches have some neighbors in reference and other images.
In one slide :) 

1. Detect keypoints
2. Triangulate a sparse set of initial matches
3. Iteratively expand matches to nearby locations
4. Use visibility constraints to filter out false matches
5. Perform surface reconstruction
Results
Results
Results
VisualSFM+PMVS
MVSNet: Depth Inference for Unstructured Multi-view Stereo

Authors: Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan
Presenter: Nail Ibrahimli
PMVS x MVSNet

Textureless Non-lambertian areas
2D CNN: quick glance

Input image → Convolutions → Pooling → Fully Connected

Convolution + ReLU + Max Pooling → Fully Connected Layer

Feature Extraction in multiple hidden layers → Classification in the output layer
Figure 3. Illustration of 3D convolution: (a) illustration of a 3D kernel to extract spatial-spectral features; (b) illustration of multiple 3D kernels to extract different kinds of spatial-spectral local feature patterns.
MVSNET

End-to-end MVS learning framework

Camera geometry encoded as differentiable homography

Variance based cost metric
Image features

8 convolutional layers

32 channel pixel descriptor

\[
\text{Images } \{I_i\}_{i=1}^{N} \xrightarrow{2D \text{ CNN}} \text{Deep Features } \{F_i\}_{i=1}^{N}
\]
Differentiable homography warping

Use intrinsic/extrinsic parameters
Warp features to the Feature volumes
Volume dimension W/4xH/4xDxF
There are N feature Volumes

\[
H_i(d) = K_i \cdot R_i \cdot \left( I - \frac{(t_i - t_i') \cdot n_i^T}{d} \right) \cdot R_i^T \cdot K_i^T
\]

Deep Features \( \{F_i\}_{i=1}^{N} \) \xrightarrow{\text{Projection Parameters}} Feature Volumes \( \{V_i\}_{i=1}^{N} \)
Cost Volume

Calculate the element wise cost of feature volumes

Dimension $W/4 \times H/4 \times D \times xF$

Feature Volumes $\{V_i\}_{i=1}^N$ \[\xrightarrow{\text{Variance}}\] Cost Volume $\mathbf{C}$

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

3D Unet Architecture

Initial dimension W/4xH/4xDxF

\[
\text{Cost Volume } C \xrightarrow{3D \text{CNN}} \text{Probability Volume } P
\]
Depth Map regression

Regressed depth based on expected value

Dimension $W/4 \times H/4 \times D \rightarrow W/4 \times H/4$

\[
D = \sum_{d=d_{\text{min}}}^{d_{\text{max}}} d \times P(d)
\]
Refine Depth map

$D \xrightarrow{2D \text{ CNN}} D_{\text{refine}}$
Loss

\begin{equation}
\text{Loss} = \sum_{p \in P_{\text{valid}}} \|d(p) - \hat{d}_i(p)\|_1 + \lambda \cdot \|d(p) - \hat{d}_r(p)\|_1
\end{equation}
MVSNET Architecture
Filtering

Photometric filtering:
\[ P(d) > 0.8 \]

Geometric filtering:
3 View visible
Filtering

(a) Inferred depth map
(b) Filtered depth map
(c) GT depth map
(d) Reference image
(e) Fused point cloud
(f) GT point cloud
Results