Fusion4D: Real-time Performance Capture of Challenging Scenes

Recent trends in 3D computer vision
Fusion4D: Real-time Performance Capture of Challenging Scenes

By:
Mingsong Dou, Sameh Khamis, Yuri Degtyarev, Philip Davidson, Sean Ryan Fanello
Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor
Pushmeet Kohli, Vladimir Tankovich, Shahram Izadi
Microsoft Research

Conference: SIGGRAPH2016
July 2016
Structure

- Related Work
- Overview
- Nonrigid Motion Field Estimation
- Fusion
- Results
- Conclusion
- Questions
Related Work

- **Offline Approach Collet**
  - 30s per frame
  - 106 cameras -> 24 depthmaps
  - controlled studio setting

- **Dynamic Fusion 2015**
  - incrementally updated Model
  - only slow motions
  - no topology-changes

- **Zollhöfer 2014**
  - Template-based
    - Fixed model
Key Contributions

- No prior
- Robust to large motion and topology changes
- Multi-view RGBD
- Real-time
Overview

• Studio: (Bild aus Teaser)
  – Trinocular cameras (2xIR + 1xRGB) 1 megapixel
  – 24 cameras -> 8 depthmaps
Overview – Setup

• Studio:
  – Trinocular cameras (2xIR + 1xRGB) 1 megapixel
  – 24 cameras -> 8 depthmaps

• Hardware:
  – 12 Computer (Intel Core i7, 3.4GHz CPU, 16GB of RAM and it uses 2 NVIDIA Titan X GPUs)
  – Master Computer (same, but with a single NVIDIA Titan X)
Overview - Pipeline

Input
- RGB
- Depth
- Segmentation

PROCESSING for current Frame
- Non-rigid alignment
- Correspondences
- ED Graph
- Alignment Error
- Volumetric Fusion & Blending
- Data Volume
- Key Volume Resampling

Output
- Key Volume
- Last key frame
- Timeline
- Rendered model
- TSDF volume

(2)
Depth Acquisition

Holoporation: Virtual 3D Teleportation in Real-time
Orts-Escolano et al. [UIST ’16]

(1)
Depth Acquisition

Real-time foreground/background segmentation

Krähenbühl et al. [ICML ’13], Vineet et al. [CVPR ’08]
Nonrigid Motion Field Estimation

- TSDF of N Depthmaps

- Embedded Deformation(ED)-model to warp the model to align with the raw depth maps

- Energy function $E(G)$ for optimization of the ED-Model
Deformation field

- (Embedded Deformation) ED-model
  - Set of K ED-nodes with sampling locations \( g_k \in \mathbb{R}^3 \) from the mesh

- Deformation (warp) of each point then is:

\[
\tilde{\tau}^p(vm;G) = \mathbb{R} \sum_{k \in s_m} w_k^m [A_k (v - g_k) + g_k + t_k] + T
\]

- and the normal \( \tilde{\tau}^n(n_m;G) = \mathbb{R} \sum_{k \in s_m} w_k^m A_k^{-T} n_m \)
Deformation field

- (Embedded Deformation) ED-model
  - Set of K ED-nodes with sampling locations \( g_k \in \mathbb{R}^3 \) from the mesh

- Deformation (warp) of each point then is:

\[
\hat{T} (v_m; G) = R \sum_{k \in s_m} w_k^m [A_k (v - g_k) + g_k + t_k] + T
\]

- and the normal \( \hat{T}^n (n_m; G) = R \sum_{k \in s_m} w_k^m A_k^{-T} n_m \)
Deformation field

- (Embedded Deformation) ED-model
  - Set of $K$ ED-nodes with sampling locations $g_k \in \mathbb{R}^3$ from the mesh

- Deformation (warp) of each point then is:

$$\mathbf{T}^p(\mathbf{v}; \mathcal{G}) = R \sum_{k \in \mathcal{S}_m} w^m_k [A_k(\mathbf{v} - g_k) + g_k + t_k] + T$$

- and the normal $\mathbf{T}^\perp(n_m; \mathcal{G}) = R \sum_{k \in \mathcal{S}_m} w^m_k A^{-T}_k n_m$
Deformation field

- (Embedded Deformation) ED-model
  - Set of $K$ ED-nodes with sampling locations $g_k \in \mathbb{R}^3$ from the mesh

- Deformation (warp) of each point then is:

$$\tilde{T}(v_m; G) = R \sum_{k \in s_m} w_k^m [A_k (v - g_k) + g_k + t_k] + T$$

- Global Rotation
- sample location
- global Translation

- node-based transform

- and the normal $\tilde{T}^\perp (n_m; G) = R \sum_{k \in s_m} w_k^m A_k^{-T} n_m$
Deformation field

- (Embedded Deformation) ED-model
  - Set of K ED-nodes with sampling locations $g_k \in \mathbb{R}^3$ from the mesh

- Deformation (warp) of each point then is:

  \[
  \mathbf{\tau}^\rho(v_m; \mathbf{G}) = \mathbf{R} \sum_{k \in s_m} w^m_k [A_k (v - g_k) + g_k + t_k] + \mathbf{T}
  \]
  weighted by distance

  \[
  \mathbf{\tau}^\perp(n_m; \mathbf{G}) = \mathbf{R} \sum_{k \in s_m} w^m_k A_k^{-T} n_m
  \]
  node-based transform

- and the normal

  December 14, 2016
  Fusion4D Slide 17
Energy Functions – Data Term

\[ E(G) = \lambda_{data} E_{data}(G) + \lambda_{hull} E_{hull}(G) + \lambda_{corr} E_{corr}(G) + \lambda_{rot} E_{rot}(G) + \lambda_{smooth} E_{smooth}(G) \]

Accumulated Misalignment:

\[ E_{data}(G) = \sum_{n=1}^{N} \sum_{m=1}^{M} \min_{x \in P(D_n)} \| \hat{T}(v_m;G) - x \|^2 \]

-> expensive
Energy Functions – Data Term

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]

Projective point-to-plane approximation of every visible point in depthmap \( N \)

\[ E_{\text{data}}(G) = \sum_{n=1}^{N} \sum_{m \in V_n(G)} (\Gamma^\perp(n_m; G)^T (\Gamma^\ast(v_m; G) - \Gamma_n(v_m; G)))^2 \]

\[ \Gamma_n(v) = P_n(\Pi_n(v)) : \]

\( \Pi_n(v) \): projection of \( v \) to a pixel in depthmap \( n \)

\( P_n(p) \): projection of pixel in depthmap \( n \) into 3D
Energy Functions – Data Term

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]

Projective point-to-plane approximation of every visible point in depthmap \( N \)

\[ E_{\text{data}}(G) = \sum_{n=1}^{N} \sum_{m \in \mathcal{V}_n(G)} \left( \mathbf{\hat{T}}(n_m; G)^T \left( \hat{\mathbf{x}}(v_m; G) - \Gamma_n(\hat{\mathbf{x}}(v_m; G)) \right) \right)^2 \]

\( \Gamma_n(v) = P_n(\Pi_n(v)) \):

Point-to-plane

Depthmap \( D_n \)
Energy Functions – Regularization Terms

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]

Restriction of the class of allowed deformations:

\[ E_{\text{rot}}(G) = \sum_{k=1}^{K} \| A_k^T A_k - I \|_F + \sum_{k=1}^{K} (\det(A_k) - 1)^2 \]

Enforce similar deformation for nearby positions:

\[ E_{\text{smooth}}(G) = \sum_{k=1}^{K} \sum_{j \in N_k} w_{jk} \rho(\| A_j (g_k - g_j) + g_j + t_j - (g_k + t_k) \|^2) \]
Energy Functions – Visual Hull Term

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]
Energy Functions – Visual Hull Term

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]

\[ E_{\text{hull}}(G) = \sum_{m=1}^{M} H(\mathbf{v}(vm;G))^2 \]

occupancy Volume: \(H(\text{voxel}) = \begin{cases} 1 & \text{voxel inside of hull} \\ 0 & \text{} \end{cases}\)

\(H(v)\): distance \(v\) to \(H\)

-> expensive: approximation of \(H\) via Gaussian blur to \(H\)
Correspondence Term – Global Patch Collider

- Decision Tree to find matches
  - 5 trees with 15 levels

- Voting Scheme:
  - Vote of all trees
    with a UNIQUE collision

- Splitfunction
  \[ f(p; \Theta) = \begin{cases} 
  L, & \text{if } I_S(p + u/D_s(p)) - I_S(p + v/D_t(p)) < \Theta \\
  R, & \text{otherwise} 
\end{cases} \]

- learned: threshold \( \Theta \), 2D Pixel offset \((u, v)\)

- Trained OFFLINE to maximize weighted harmonic mean between precision and recall
  - Ground truth: accurate Method of [Dou et al. 2015]
Correspondence Term - Global Patch Collider
Energy Functions – Correspondence Term

\[ E(G) = \lambda_{\text{data}} E_{\text{data}}(G) + \lambda_{\text{hull}} E_{\text{hull}}(G) + \lambda_{\text{corr}} E_{\text{corr}}(G) + \lambda_{\text{rot}} E_{\text{rot}}(G) + \lambda_{\text{smooth}} E_{\text{smooth}}(G) \]

Set of \( F_n \) Matches \( \{u_{nf}^{\text{prev}}, u_{nf}\} \)

Corresponding point \( q_{nf} \in \mathbb{R}^3 \) per match:

\[ q_{nf} = \underset{v \in V}{\text{argmin}} \left\| \Pi_n (\mathbf{r}(v, G^{\text{prev}})) - u_{nf}^{\text{prev}} \right\| \]

Correspondence Term:

\[ E_{\text{corr}}(G) = \sum_{n=1}^{N} \sum_{f=1}^{F_n} \rho \left( \| \mathbf{r}(q_{nf}; G) - p_n(u_{nf}) \|^2 \right) \]

- Deformed point
- 3D correspondence
- robustifier
Optimization of the ED-Model

Every single Energy function was squared, $E(G) = f(X)^T f(X)$ possible
- standard sparse linear least squares minimization problem

- Initialization:
  - ED-nodebased transformation of the old ED-Model fixed $\{A_k, t_k\}_{k=1}^K$
  - Iterative closest Points (4 Iterations) for the global translation and rotation $\{T, R\}$
Optimization of the ED-Model

• Levenberg Marquardt Algorithm: damping factor

\[(J^T J + \mu I) h = -J^T f\]

• Stepwise update:
  – Accepted if \(E(f(X + h)) < E(X)\), \(\mu\) lowered -> more aggressive
  – Otherwise \(\mu\) increased and solved again

Discontinuities in \(E_{\text{data}}(G)\):
\[\Gamma_n(I'(v_m; G))\]

Differentiable Approximation:
\[\sum_{n=1}^{N} \sum_{m \in V_n(G_0)} (n \tilde{m}(G_0)^T (\tilde{v}_m(G) - \Gamma_n(\tilde{v}_m((G_0))))^2\]

current Parameters
Linear Solving

- Store $J^T J$ and $J^T f$ instead of $J$, since it is magnitudes larger
  - Lower memory footprint -> time efficiency
  - $E_{\text{data}}(G)$ Approximation -> independent of the number of cameras
  - $J^T J$ is a sparse matrix -> consists of nonzero blocks
    » Blockwise calculation via CUDA blocks

- iterative solve of $J^T J$ via preconditioned conjugate gradient (PCG)
  - Preconditioner: diagonal Blocks of $J^T J$

- DynamicFusion 2015 uses direct sparse Cholesky decomposition:
  - Approximation of $J^T J$
Overview - Pipeline

Input
- RGB
- Depth
- Segmentation

PROCESSING for current Frame
- Non-rigid alignment
  - Correspondences
  - ED Graph
  - Alignment Error

Data Volume
- Volumetric Fusion & Blending

Output
- Key Volume
- TSDF volume
- Rendered model

Timeline
- Current
- last key frame

(1)
Fusion

• Fusion at Data Frame
  – Warping
  – Selective Fusion
  – Blending

• Fusion at Reference Frame

• Key Volume
Volume Warping

- Warp key Volume to align the data via ED-model
  - Voxel ∈ ℝ³ containing ⟨d, w⟩ in the key Volume

- Fusion into fused TSDF: \( x^d \) weighted average of warped neighboring reference-voxels \( \langle \tilde{d}^r, \tilde{w}^r \rangle \)

Correction of the depthvalue:
\[
\tilde{d}^r = d^r + (\tilde{x}^r - x^d)^T \tilde{\Delta}
\]
Fusion at Data frame – Selective Fusion

- **Voxel Collision**
  - $|x - \hat{x}^R| > \eta$
  - $\hat{x}^R$ closest reference voxel

- **Voxel Misalignment**
  - Discard voxel if alignment error $> \text{threshold}$

\[
e_{\tilde{x}^r} = \begin{cases} 
|D^d(\tilde{x}^r)|, & \text{fused TSDF} \\
\min(|D^d(\tilde{x}^r)|, \mathcal{H}^d(\tilde{x}^r)), & \text{Hull}
\end{cases}
\]

(3) reference volume, without and with voxelcolldetect.
Fusion at Data frame - Blending

- Fuse dataVolume and warped reference Volume
  - No naive blending (boundary-voxels)

\[ \bar{d}^d = \frac{\tilde{d}^r \tilde{w}^r (1.0 - e_{voxel}) + d^d w^d}{\tilde{w}^r (1.0 - e_{voxel}) + w^d} \]

\( e_{voxel} \): aggregated average of \( e_{pixel} \) of every depthmap

\[
e_{pixel} = \begin{cases} 
\min \left( 1.0, \frac{|d - d_{proj}|}{d_{max}} \right), & \text{if } d_{proj} \text{ is valid} \\
1.0, & \text{otherwise}
\end{cases}
\]
Fusion at the Reference Frame

Just like DynamicFusion:

- Discarding Voxels not aligned well to the Data, and refreshing them
  - $alignmenterror e_{xr}$
- Update depth and weight by projecting it to the depthmaps

BUT: Key Volumes

- Complete Reset periodically (here 10 frames)
Key Volumes

Live point cloud

Key Volume

(1)
Key Volumes

- Live point cloud
- TSDF Volume

align

(1)
Key Volumes

Live point cloud

TSD Volume

Key Volume

update

(1)
Key Volumes
Key Volumes

Live point cloud

TSDF Volume

Key Volume

...
Key Volumes

Live point cloud → TSDF Volume → Key Volume → ...

(1)
Key Volumes

Live point cloud → TSDF Volume → Key Volume → ...
Key Volumes
Conclusion

- Time Consumption:

12 Computer´s

<table>
<thead>
<tr>
<th>Step</th>
<th>Master Computer</th>
<th>12 Computer´s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. stereo matching</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>2. segmentation</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3. correspondence estimation</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>1. preprocessing</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2. rigid pose estimation</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3. nonrigid registration</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4. fusion</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

parallel

December 14, 2016 Slide 45
Results

- robustness to fast motion

- complex topology changes

(3)
Comparison

- Our method
- Newcombe et al. 2015
- Zollhöfer et al. 2014
- GPC (ours)
- SIFT
- FAST+SIFT
- DeepMatch
- EpicFlow
Limitations

- Lost tracking
- Segmentation error
- Alignment error
Sources

  Source Materials „Supplemental files“
• (3): Fusion4D: Real-time Performance Capture of Challenging Scenes
  in SIGGRAPH2016
• (4): Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration
  07.12.2016
• (5): The Global Patch Collider in CVPR
Questions?