IsMo-GAN: Adversarial Learning for Monocular Non-Rigid 3D Reconstruction

Soshi Shimada 1,2, Vladislav Golyanik 2,3, Christian Theobalt 3, and Didier Stricker 1,2

1. Augmented Vision, DFKI        2. University of Kaiserslautern        3. MPI for Informatics
Motivation
Motivation

- 3D reconstruction of a deformable object from monocular 2D image sequences is still a challenging problem
Motivation

- 3D reconstruction of a deformable object from monocular 2D image sequences is still a challenging problem
Related works
Related works

- Non Rigid Structure from Motion (NRSfM)
Related works

• Non Rigid Structure from Motion (NRSfM)
  - input: point tracks on 2D frames

Figure 1. NRSfM technique (Golyanik et al., 2017)
Related works

• Non Rigid Structure from Motion (NRSfM)
  - input: point tracks on 2D frames
  - basically no limitation regarding target objects

Figure 1. NRSfM technique (Golyanik et al., 2017)
Related works

- Non Rigid Structure from Motion (NRSfM)
  - input: point tracks on 2D frames
  - basically no limitation regarding target objects
  - multiple frames are required
Related works

- Non Rigid Structure from Motion (NRSfM)
  - input: point tracks on 2D frames
  - basically no limitation regarding target objects
  - multiple frames are required
  - difficulty to apply on non-textured objects

Figure 1. NRSfM technique (Golyanik et al., 2017)
Related works
Related works

- Template based

*Figure 2. 3D reconstruction from a sequence of images (Yu et al., 2015)*
Related works

• Template based
  - input: 3D template & 2D images

Figure 2. 3D reconstruction from a sequence of images (Yu et al, 2015)
Related works

• Neural network based
Related works

• Neural network based
  - Input: a single/sequence of images
Related works

• Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
Related works

• Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
  - E.g. HDM-Net
Related works

• Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
  - E.g. HDM-Net

Figure 3. HDM-Net (Golyanik, 2018)
Related works

- Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
  - E.g. HDM-Net

Figure 3. HDM-Net (Golyanik, 2018)

- 3D Reconstruction from a single RGB image
Related works

- Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
  - E.g. HDM-Net

- 3D Reconstruction from a single RGB image
- Regress 3D coordinates (xyz geometry)

*Figure 3. HDM-Net (Golyanik, 2018)*
Related works

- Neural network based
  - Input: a single/sequence of images
  - Output: 3D geometry (Voxel/Point Set/Mesh)
  - E.g. HDM-Net

Figure 3. HDM-Net (Golyanik, 2018)

- 3D Reconstruction from a single RGB image
- Regress 3D coordinates (xyz geometry)
- Apply 2D conv. not 3D conv.
Limitations of HDM-Net
Limitations of HDM-Net

- In a real-world scenario, our architecture has difficulty to reconstruct geometries especially when
Limitations of HDM-Net

• In a real-world scenario, our architecture has difficulty to reconstruct geometries especially when

  1) the deformation states in the scene is quite different from the ones in the training dataset
Limitations of HDM-Net

- In a real-world scenario, our architecture has difficulty to reconstruct geometries especially when
  1) the deformation states in the scene is quite different from the ones in the training dataset
  2) the scene has a complicated background
Overview

Input
2D Image

Point Cloud Generator

\( L_{iso.} \)
\( L_{3D} \)
\( L_{adv.} \)

3D Output

Discriminator

GT
Overview

2D Image → Point Cloud Generator → 3D Output

\[ L_{iso.} \rightarrow L_{3D} \rightarrow L_{adv.} \]

Discriminator

GT
Overview
Overview

IsMo-GAN (Point Cloud Generator)
Overview

OD-Net

IsMo-GAN (Point Cloud Generator)
Overview

OD-Net

Confidence Map

IsMo-GAN (Point Cloud Generator)

Input 2D Image
Overview

Input
2D Image

OD-Net

Confidence Map

Binarisation
&
Contour fill

Binary Mask

IsMo-GAN (Point Cloud Generator)
Overview

Input 2D Image

OD-Net -> Confidence Map -> Binarisation & Contour fill -> Binary Mask

IsMo-GAN (Point Cloud Generator)
Overview

OD-Net

Confidence Map

Binarisation & Contour fill

Binary Mask

Masked-out Input

IsMo-GAN (Point Cloud Generator)

Input 2D Image
Overview

IsMo-GAN (Point Cloud Generator)

OD-Net → Confidence Map → Binarisation & Contour fill → Binary Mask

Masked-out Input → Rec-Net
IsMo-GAN (Point Cloud Generator)

OD-Net

Confidence Map

Binarisation & Contour fill

Binary Mask

Masked-out Input

Rec-Net

Output 3D Geometry

Input 2D Image
Overview

Input
2D Image

Point Cloud Generator

3D Output

Discriminator

\[ \mathcal{L}_{\text{iso.}} \quad \mathcal{L}_{3D} \quad \mathcal{L}_{\text{adv.}} \]
Overview

Input
2D Image

Point Cloud Generator

3D Output

Discriminator

$\mathcal{L}_{\text{iso.}} \quad \mathcal{L}_{3D} \quad \mathcal{L}_{\text{adv.}}$

GT
Loss functions
Loss functions

• In order to penalize the network output critically, three kinds of loss functions were incorporated.
Loss functions

• In order to penalize the network output critically, three kinds of loss functions were incorporated.

1) 3D error
2) Isometry prior
3) Adversarial loss
Loss functions

1) 3D error
Loss functions

1) 3D error

\[ MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{X}_i - X_i)^2 \]

- Main loss component for 3D coordinates regression
Loss functions

1) 3D error

\[ MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{X}_i - X_i)^2 \]

- Main loss component for 3D coordinates regression
- Penalize difference between 3D coordinates of output and GT
Loss functions

2) Isometry prior
Loss functions

2) Isometry prior

- Idea: since we assume our target object is isometric, a vertex position has to be close to neighboring vertices.
2) Isometry prior

- Idea: since we assume our target object is isometric, a vertex position has to be close to neighboring vertices.
2) Isometry prior

- Idea: since we assume our target object is isometric, a vertex position has to be close to neighboring vertices.
2) Isometry prior

- Idea: since we assume our target object is isometric, a vertex position has to be close to neighboring vertices.

- Apply gaussian smoothing on the output and compute the difference between $X_G$ and $X$. 

Loss functions
2) Isometry prior

- Idea: since we assume our target object is isometric, a vertex position has to be close to neighboring vertices.

- Apply gaussian smoothing on the output and compute the difference between $X_G$ and $X$.

$$X_G = \frac{1}{2\pi\sigma^2} exp \left( -\frac{x^2 + y^2}{2\sigma^2} \right) \ast X$$

$$SAD = \sum_{i=1}^{n} |X_G - X_i|$$
Loss functions

3) Adversarial loss
3) Adversarial loss

Loss functions
3) Adversarial loss

- For further generalisability, the network is trained in an adversarial manner
Overview

Input
2D Image → Point Cloud Generator → 3D Output

\( \mathcal{L}_{iso.} \) \( \mathcal{L}_{3D} \) \( \mathcal{L}_{adv.} \)

Discriminator

GT
Overview

Input 2D Image → Point Cloud Generator → 3D Output

\( \mathcal{L}_{\text{iso.}} \) \( \mathcal{L}_{3D} \) \( \mathcal{L}_{\text{adv.}} \)

Discriminator

GT
Datasets
Datasets

- Generated 4648 deformation states on blender game engine
Datasets

- Generated 4648 deformation states on blender game engine
- Took 5 images for each state.
Datasets

- Generated 4648 deformation states on blender game engine
- Took 5 images for each state.
- 4 different textures (Organ, Flag, Cloth, Carpet)
Datasets

- Generated 4648 deformation states on blender game engine
- Took 5 images for each state.

- 4 different textures (Organ, Flag, Cloth, Carpet)
- 5 different illumination positions
Datasets

- Generated 4648 deformation states on blender game engine
- Took 5 images for each state.

- 4 different textures (Organ, Flag, Cloth, Carpet)
- 5 different illumination positions
- 4648 ply file and 330K images in total (4648x5x4x3 + 4648x5 + 4648x5)
Evaluation and Visualization
## Quantitative Results

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$t$, sec.</td>
<td>3.305</td>
<td>5.42</td>
<td>0.035</td>
<td>0.39</td>
<td>0.005</td>
<td>0.004</td>
</tr>
<tr>
<td>$e_{3D}$</td>
<td>1.3258</td>
<td>1.0049</td>
<td>1.6189</td>
<td>0.46</td>
<td>0.0251</td>
<td>0.0175</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>0.007</td>
<td>0.0176</td>
<td>1.23</td>
<td>0.0334</td>
<td>0.03</td>
<td>0.01</td>
</tr>
</tbody>
</table>

**Table 1:** Reconstruction times per frame $t$ in seconds, $e_{3D}$ and standard deviation $\sigma$ for Yu et al. [71], Liu-Yin et al. [38], AMP [18], VA [15], HDM-Net [17] and our IsMo-GAN method, for the test interval of 400 frames.

<table>
<thead>
<tr>
<th></th>
<th>illum. 1</th>
<th>illum. 2</th>
<th>illum. 3</th>
<th>illum. 4</th>
<th>illum. 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDM-Net [17]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$e_{3D}$</td>
<td>0.07952</td>
<td>0.0801</td>
<td>0.07942</td>
<td>0.07845</td>
<td>0.07827</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>0.0525</td>
<td>0.0742</td>
<td>0.0888</td>
<td>0.1009</td>
<td>0.1123</td>
</tr>
<tr>
<td>IsMo-GAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$e_{3D}$</td>
<td>0.06803</td>
<td>0.06908</td>
<td>0.06737</td>
<td>0.06754</td>
<td>0.06685</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>0.0499</td>
<td>0.0696</td>
<td>0.0824</td>
<td>0.093</td>
<td>0.102</td>
</tr>
</tbody>
</table>

**Table 2:** Comparison of 3D error for different illuminations. The illuminations 1-4 are known, and the illumination 5 is unknown.
Different textures

<table>
<thead>
<tr>
<th>Model</th>
<th>$e_{3D}$</th>
<th>$\sigma$</th>
<th>$e_{3D}$</th>
<th>$\sigma$</th>
<th>$e_{3D}$</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDM-Net [17]</td>
<td>0.0485</td>
<td>0.0135</td>
<td>0.0499</td>
<td>0.022</td>
<td>0.0489</td>
<td>0.0264</td>
</tr>
<tr>
<td>IsMo-GAN</td>
<td>0.0336</td>
<td>0.0148</td>
<td>0.0333</td>
<td>0.0208</td>
<td>0.0353</td>
<td>0.0242</td>
</tr>
</tbody>
</table>

Table 3: $e_{3D}$ comparison for differently textured surfaces under the same illumination (illumination 1).
Real-world images

(a) (b) (c) (d) (e) (f)

Input

Mask applied (OD-Net)

Output (IsMo-GAN)

Output (HDM-Net)
Origami sequences
Real texture-less cloth

Texture less dataset (Bednarik et al., 2018)

Input (real)

IsMo-GAN (ours)

HDM-Net
Real texture-less cloth
Conclusion
Conclusion

• IsMo-GAN excels other model-based approaches in accuracy and inference time (250 HZ)
Conclusion

- IsMo-GAN excels other model-based approaches in accuracy and inference time (250 HZ)
- Robust to illumination position changes
Conclusion

• IsMo-GAN excels other model-based approaches in accuracy and inference time (250 HZ)
• Robust to illumination position changes
• Thanks to OD-Net, IsMo-GAN shows better generalizability in a texture-less and real-world scenario comparing with HDM-Net
Conclusion

- IsMo-GAN excels other model-based approaches in accuracy and inference time (250 HZ).
- Robust to illumination position changes.
- Thanks to OD-Net, IsMo-GAN shows better generalizability in a texture-less and real-world scenario comparing with HDM-Net.
- (Limitation) Training data (deformation state) is limited.
References

Thank you for your attention!
Texture-less dataset
• Extract 20 sequential deformation from each 100 states as a test dataset

• Training:Test = 8 : 2
External Occlusion

Input

IsMo-GANs (ours)

HDM-Net

R=1
R=13
R=25
R=37

GT