Photorealistic image synthesis for object instance detection

Tomas Hodan, Vibhav Vineet, Ran Gal, Emanuel Shalev, Jon Hanzelka, Treb Connell, Pedro Urbina, Sudipta N. Sinha, Brian Guenter

International Conference on Image Processing (ICIP) 2019
September 23th, Taipei
CNN’s are great, but data hungry

Large amounts of annotated training images required.
CNN’s are great, but data hungry

Large amounts of annotated training images required.

Expensive to annotate real images.

Image classification $\$\
Object detection $\$$\
6D object pose estimation $\$$\$$
CNN’s are great, but data hungry

Large amounts of annotated training images required.

Expensive to annotate **real images**.

Training with **synthetic images**?
CNN’s are great, but data hungry

Large amounts of annotated training images required.

Expensive to annotate real images.

Training with synthetic images?
Scales well as only minimal human effort is required.
Common approaches to synthesize training images

Approach 1: Cut & paste on photographs

Object segments cut from real images  
Background photographs
Common approaches to synthesize training images

Approach 1: Cut & paste on photographs

Object segments cut from real images  Background photographs

Object detection
Dwibedi ICCV’17, Dvornik ECCV’18

6D object pose estimation
Rad ICCV’17, Tekin CVPR’18
Common approaches to synthesize training images

Approach 2: **Rendering 3D object models on photographs**

3D object models

→

Background photographs
Common approaches to synthesize training images

Approach 2: **Rendering 3D object models on photographs**

- **Object detection**
  - Hinterstoisser ICCVW’19

- **Viewpoint estimation**
  - Su ICCV’15

- **Optical flow estimation**
  - Dosovitskiy ICCV’15
Problem: lack of photorealism

Inconsistent lighting of the objects and the background scene.

Missing interreflections and shadows.

Unnatural object pose and context.
Problem: lack of photorealism

Inconsistent lighting of the objects and the background scene.

Missing interreflections and shadows.

Unnatural object pose and context.

→ Domain gap between the synthetic and real images.
Problem: lack of photorealism

Inconsistent lighting of the objects and the background scene.

Missing interreflections and shadows.

Unnatural object pose and context.

→ Domain gap between the synthetic and real images.

→ Low performance on real when trained only on synthetic.

Su ICCV’15: Render for CNN: viewpoint estimation in images using CNNs trained with...
Richter ECCV’16: Playing for data: Ground truth from computer games.
Rozantsev TPAMI’18: Beyond sharing weights for deep domain adaptation.
Reducing the domain gap

**Domain adaptation (DA):** Learning domain invariant features or transferring models from one domain to another (Csurka’17).
Reducing the domain gap

**Domain adaptation (DA):** Learning domain invariant features or transferring models from one domain to another (Csurka’17).

**Photorealistic rendering:** Presumably complementary to DA.
Reducing the domain gap

**Domain adaptation (DA):** Learning domain invariant features or transferring models from one domain to another (Csurka’17).

**Photorealistic rendering:** Presumably complementary to DA.

a) **Rasterization techniques** - e.g. OpenGL, DirectX

Viewpoint estimation
Attias ECCV’16

6D object pose estimation
Tremblay CoRL’18
Reducing the domain gap

**Domain adaptation (DA):** Learning domain invariant features or transferring models from one domain to another (Csurka’17).

**Photorealistic rendering:** Presumably complementary to DA.

a) **Rasterization techniques** - e.g. OpenGL, DirectX

![Viewpoint estimation](image1)

Attias ECCV’16

6D object pose estimation

Tremblay CoRL’18

b) **Physically based rendering (PBR)** - e.g. Arnold, Mitsuba

![Gaze estimation](image2)

(Wood ICCV’15)

![Segmentation](image3)

Segmentation, normal estimation, boundary detection

(Zhang CVPR’17)

![Intrinsic image decomposition](image4)

Li ECCV’18
Rendering techniques

**Rasterization** - e.g. OpenGL, DirectX

- ✔️ Fast (multiple VGA frames per second).
- ❌ Custom shaders to approximate complex illumination effects (scattering, refraction and reflection) yield difficult-to_eliminate artifacts.
Rendering techniques

**Rasterization** - e.g. OpenGL, DirectX

- ✔ Fast (multiple VGA frames per second).
- ✗ Custom shaders to approximate complex illumination effects (scattering, refraction and reflection) yield difficult-to-eliminate artifacts.

**Physically based rendering** - e.g. Arnold, Mitsuba

- ✔ Ray tracing to accurately simulate complex illumination effects.
- ✔ Highly realistic images, difficult to distinguish from real images.
- ✗ Slow (may take multiple minutes per VGA frame).
How effective is PBR for training an object detector?
How effective is PBR for training an object detector?

The proposed approach for synthesis of training images:

1. **3D object models rendered in 3D models of scenes** with realistic PBR materials and lighting.
2. **Plausible geometric configuration** of objects and cameras in a scene generated using physics simulation.
3. **High photorealism** of the synthesized images achieved by PBR.
How effective is PBR for training an object detector?

The proposed approach for synthesis of training images:

1. **3D object models rendered in 3D models of scenes** with realistic PBR materials and lighting.
2. **Plausible geometric configuration** of objects and cameras in a scene generated using physics simulation.
3. **High photorealism** of the synthesized images achieved by PBR.

Applicable to other object-centric tasks such as instance segmentation and 6D object pose estimation.
Scene and object modeling

**3D scene models:** Indoor scenes with PBR materials.

1. Reconstructions of real scenes (using LIDAR, photogrammetry, 3D scans, PBR material scanning)
2. Purchased online
3. Shelf from APC with assigned PBR materials
Scene and object modeling

**3D scene models:** Indoor scenes with PBR materials.

- Reconstructions of real scenes (using LIDAR, photogrammetry 3D scans, PBR material scanning)
- Purchased online
- Shelf from APC with assigned PBR materials

**3D object models:** From Linemod and Rutgers APC datasets with assigned PBR materials.

- Linemod objects (rendered in scenes 1-5)
- Rutgers APC objects (rendered in scene 6)
Scene and object composition

**Stages for objects:** Manually defined polygons on scene surfaces (tables, chairs, etc.) to place the objects on.
Scene and object composition

**Stages for objects:** Manually defined polygons on scene surfaces (tables, chairs, etc.) to place the objects on.

**Generating object arrangements:**
1. Poses of the object models are instantiated above a stage.
2. Physically plausible poses are reached using physics simulation.
Scene and object composition

**Stages for objects:** Manually defined polygons on scene surfaces (tables, chairs, etc.) to place the objects on.

**Generating object arrangements:**
1. Poses of the object models are instantiated above a stage.
2. Physically plausible poses are reached using physics simulation.

**Camera positioning:** Multiple cameras are positioned around each object arrangement.
Physically based rendering

Rendered on a CPU cluster with 400 nodes (16-core processors).
Physically based rendering

Rendered on a CPU cluster with 400 nodes (16-core processors).

**PBR images of 3 quality settings** rendered from each camera:
1. **Low**: ~15s per image, 2.3M images per day.
2. **Medium**: ~120s per image, 288K images per day.
3. **High**: ~720s per image, 48K images per day.
Examples of rendered images
Examples of rendered images

A dataset of 400K PBR images available at: [thodan.github.io/objectsynth](thodan.github.io/objectsynth)

Each object instance annotated with a 2D bounding box, a segmentation mask and a 6D pose.
Experiments: Datasets

Linemod-Occluded (Hinterstoisser ACCV’12, Brachmann ECCV’14)
Experiments: Datasets

Linemod-Occluded (Hinterstoisser ACCV’12, Brachmann ECCV’14)

Rutgers APC (Rennie RAL’16)
Experiments: Baseline training images (BL)

Object models rendered (OpenGL) on random photographs, as in Hinterstoisser ECCVW’18.
Experiments: Baseline training images (BL)

Object models rendered (OpenGL) on random photographs, as in Hinterstoisser ECCVW'18.

Object models rendered in the same poses as in the PBR images.
## Experiments: Importance of PBR images

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Architecture</th>
<th>PBR-h</th>
<th>PBR-l</th>
<th>PBR-ho</th>
<th>BL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-O</td>
<td>Inc.-ResNet-v2</td>
<td>55.9</td>
<td>49.8</td>
<td>–</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>49.9</td>
<td>44.6</td>
<td>–</td>
<td>45.1</td>
</tr>
<tr>
<td>RU-APC</td>
<td>Inc.-ResNet-v2</td>
<td>71.9</td>
<td>72.9</td>
<td>58.7</td>
<td>48.0</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>68.4</td>
<td>65.1</td>
<td>51.6</td>
<td>52.7</td>
</tr>
</tbody>
</table>

Performance (mAP@.75IoU) of Faster R-CNN (Ren NIPS’15).

**High-quality PBR** images outperform **BL** images by 5-11% on Linemod-Occluded and 16-24% on Rutgers APC.
Experiments: Importance of PBR quality

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Architecture</th>
<th>PBR-h</th>
<th>PBR-l</th>
<th>PBR-ho</th>
<th>BL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-O</td>
<td>Inc.-ResNet-v2</td>
<td>55.9</td>
<td>49.8</td>
<td>–</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>49.9</td>
<td>44.6</td>
<td>–</td>
<td>45.1</td>
</tr>
<tr>
<td>RU-APC</td>
<td>Inc.-ResNet-v2</td>
<td>71.9</td>
<td>72.9</td>
<td>58.7</td>
<td>48.0</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>68.4</td>
<td>65.1</td>
<td>51.6</td>
<td>52.7</td>
</tr>
</tbody>
</table>

Performance (mAP@.75IoU) of Faster R-CNN (Ren NIPS’15).

High-quality PBR images outperform low-quality PBR images by 5-6% on Linemod-Occluded.
Experiments: Importance of PBR quality

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Architecture</th>
<th>PBR-h</th>
<th>PBR-l</th>
<th>PBR-ho</th>
<th>BL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-O</td>
<td>Inc.-ResNet-v2</td>
<td>55.9</td>
<td>49.8</td>
<td>–</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>49.9</td>
<td>44.6</td>
<td>–</td>
<td>45.1</td>
</tr>
<tr>
<td>RU-APC</td>
<td>Inc.-ResNet-v2</td>
<td>71.9</td>
<td>72.9</td>
<td>58.7</td>
<td>48.0</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>68.4</td>
<td>65.1</td>
<td>51.6</td>
<td>52.7</td>
</tr>
</tbody>
</table>

Performance (mAP@.75IoU) of Faster R-CNN (Ren NIPS’15).

High-quality PBR images outperform low-quality PBR images by 5-6% on Linemod-Occluded.

No significant improvement on Rutgers APC objects rendered in the simpler scene 6. → The low PBR quality is sufficient for scenes with simpler illumination and materials.
# Experiments: Importance of scene context

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Architecture</th>
<th>PBR-h</th>
<th>PBR-l</th>
<th>PBR-ho</th>
<th>BL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-O</td>
<td>Inc.-ResNet-v2</td>
<td>55.9</td>
<td>49.8</td>
<td>–</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>49.9</td>
<td>44.6</td>
<td>–</td>
<td>45.1</td>
</tr>
<tr>
<td>RU-APC</td>
<td>Inc.-ResNet-v2</td>
<td>71.9</td>
<td>72.9</td>
<td>58.7</td>
<td>48.0</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>68.4</td>
<td>65.1</td>
<td>51.6</td>
<td>52.7</td>
</tr>
</tbody>
</table>

Performance (mAP@.75IoU) of Faster R-CNN (Ren NIPS’15).

RU-APC objects rendered in **two setups:**

1) In context (PBR-h)  
2) Out of context (PBR-ho)  
Example real test image
Experiments: Importance of scene context

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Architecture</th>
<th>PBR-h</th>
<th>PBR-l</th>
<th>PBR-ho</th>
<th>BL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM-O</td>
<td>Inc.-ResNet-v2</td>
<td>55.9</td>
<td>49.8</td>
<td>–</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>49.9</td>
<td>44.6</td>
<td>–</td>
<td>45.1</td>
</tr>
<tr>
<td>RU-APC</td>
<td>Inc.-ResNet-v2</td>
<td>71.9</td>
<td>72.9</td>
<td>58.7</td>
<td>48.0</td>
</tr>
<tr>
<td></td>
<td>ResNet-101</td>
<td>68.4</td>
<td>65.1</td>
<td>51.6</td>
<td>52.7</td>
</tr>
</tbody>
</table>

Performance (mAP@.75IoU) of Faster R-CNN (Ren NIPS’15).

RU-APC objects rendered in **two setups**:

1) **In context** (PBR-h)  
2) **Out of context** (PBR-ho)

Example real test image

**In context** images outperform **out of context** images by **13-16%**.
Conclusions

**Faster R-CNN achieves 5–24% higher mAP@.75IoU** on real test images when trained on photorealistic images synthesized by the proposed approach.
Conclusions

Faster R-CNN achieves 5–24% higher mAP@.75IoU on real test images when trained on photorealistic images synthesized by the proposed approach.

Low PBR quality is sufficient in scenes with simple illumination and materials.
Conclusions

Faster R-CNN achieves 5–24% higher mAP@.75IoU on real test images when trained on photorealistic images synthesized by the proposed approach.

Low PBR quality is sufficient in scenes with simple illumination and materials.

Accurately modeling context of the test scene helps.
Conclusions

Faster R-CNN achieves 5–24% higher mAP@.75IoU on real test images when trained on photorealistic images synthesized by the proposed approach.

Low PBR quality is sufficient in scenes with simple illumination and materials.

Accurately modeling context of the test scene helps.

A public dataset of 400K PBR images available at: thodan.github.io/objectsynth