Deep 3D Pose Regression of Real Objects Trained With Synthetic Data

Author:
Pau Bramon Mora

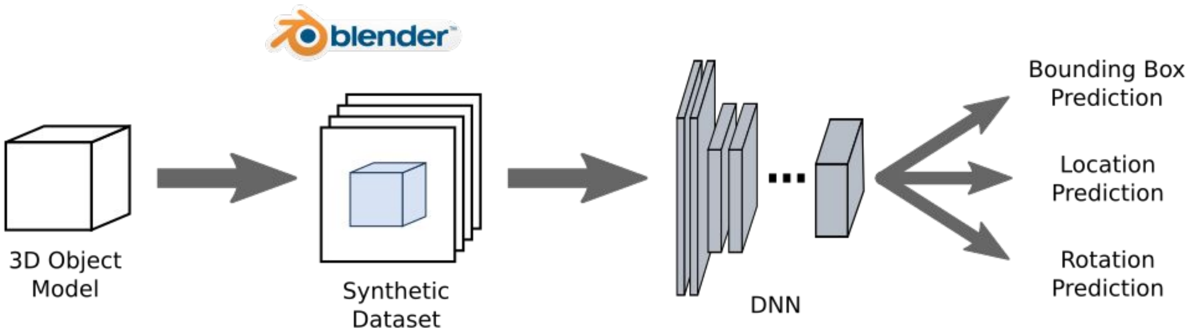
Supervisor:
Sergio Escalera Guerrero

Outline

1. Overview

Regress the location and rotation of objects from RGB images:

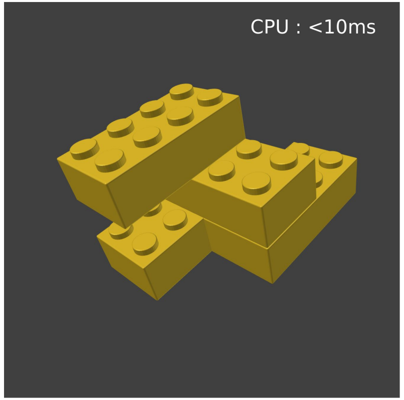
- Training Deep Models exclusively with synthetic images
- Using a multi-task Neural Network architecture with a novel loss called *Projection Loss*

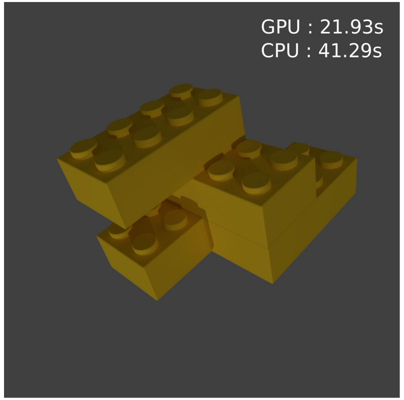


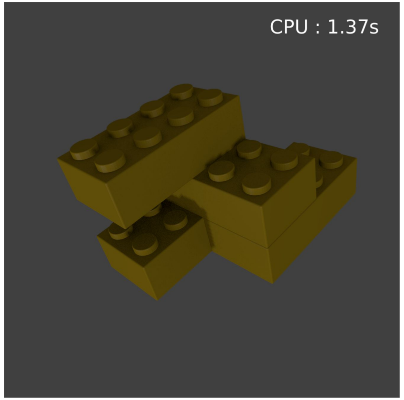
2.1. Synthetic Generation |

Main problems when using synthetic images for training:

- Reality Gap
- Designing realistic scenes is tedious work
- Rendering is computationally expensive



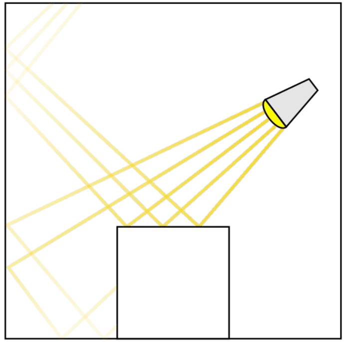


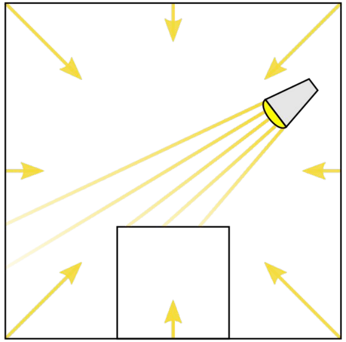


2.1. Synthetic Generation II

Domain Randomization:

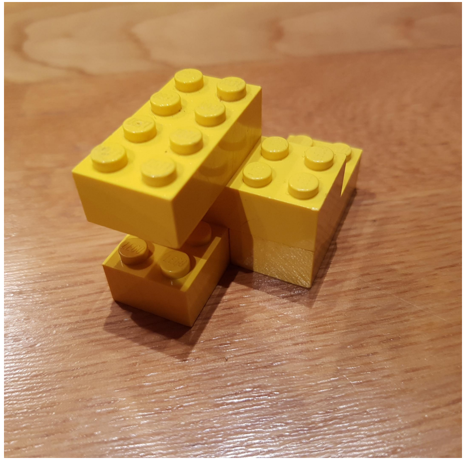
- Background randomization
- Object randomization
- Light simplification and randomization





2.2. Real dataset

A real dataset with three objects for test:

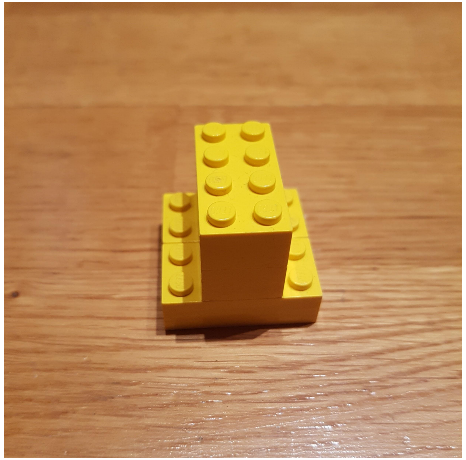


Toy

Object with a complex shape. Strange views with many different shades.

Box

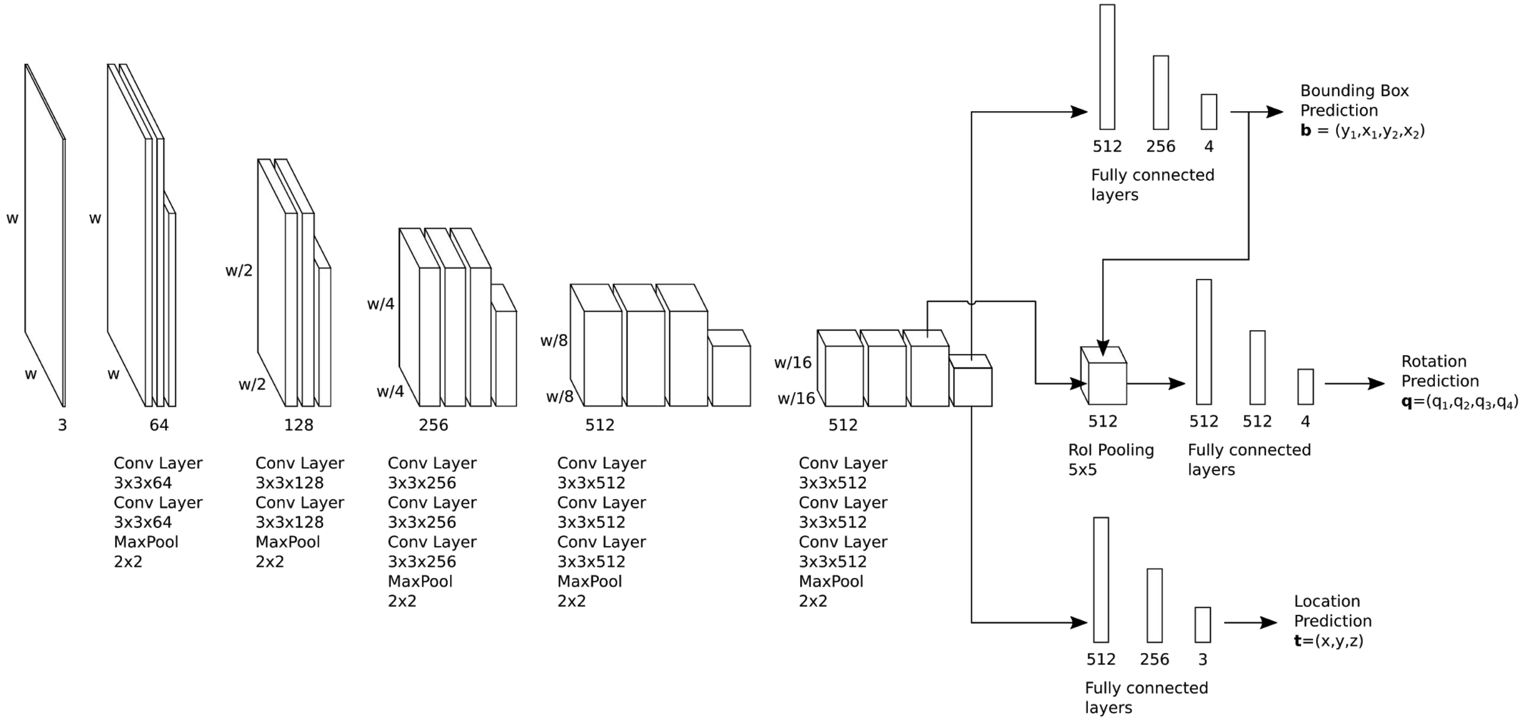
Box with serigraph. Difficult to generate in Blender.



Symmetric

Symmetric object. Problematic for the optimization process.

3.1. Network architecture I



3.1. Network architecture II

Bounding Box Loss

$$\mathcal{L}_b = \|\tilde{\mathbf{b}} - \mathbf{b}\|$$

Location Loss

$$\mathcal{L}_t = \|\tilde{\mathbf{t}} - \mathbf{t}\|$$

Rotation Loss

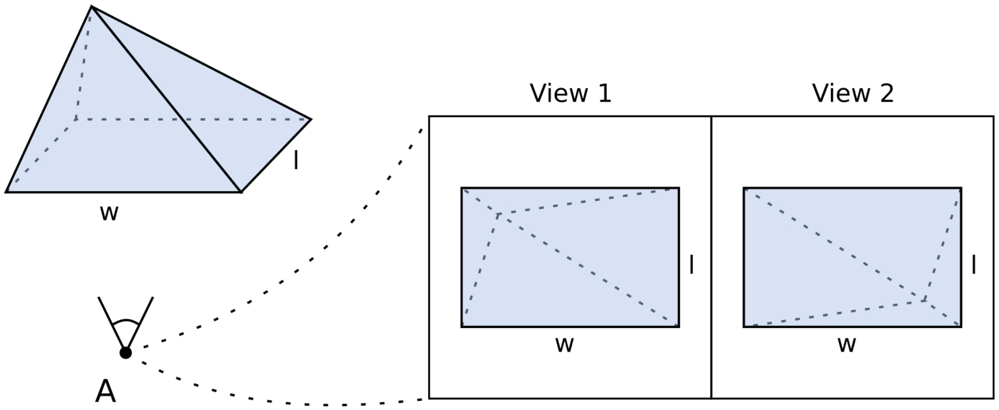
$$\mathcal{L}_r = \frac{1}{P} \sum_{\mathbf{x} \in \mathcal{M}} \|\mathbf{R}\mathbf{x} - \tilde{\mathbf{R}}\mathbf{x}\|$$

$$\boxed{\mathcal{L}_{total} = \lambda_b \mathcal{L}_b + \lambda_q \mathcal{L}_q + \lambda_t \mathcal{L}_t + \lambda_{PL} \mathcal{L}_{PL} + \lambda_{reg} \mathcal{L}_{reg}}$$

3.2. Projection Loss |

Projection Loss to deal with view ambiguities and symmetries

- In these cases, the network should predict a valid solution
- The silhouettes of the two objects should match
- IoU as a measure to compare silhouettes



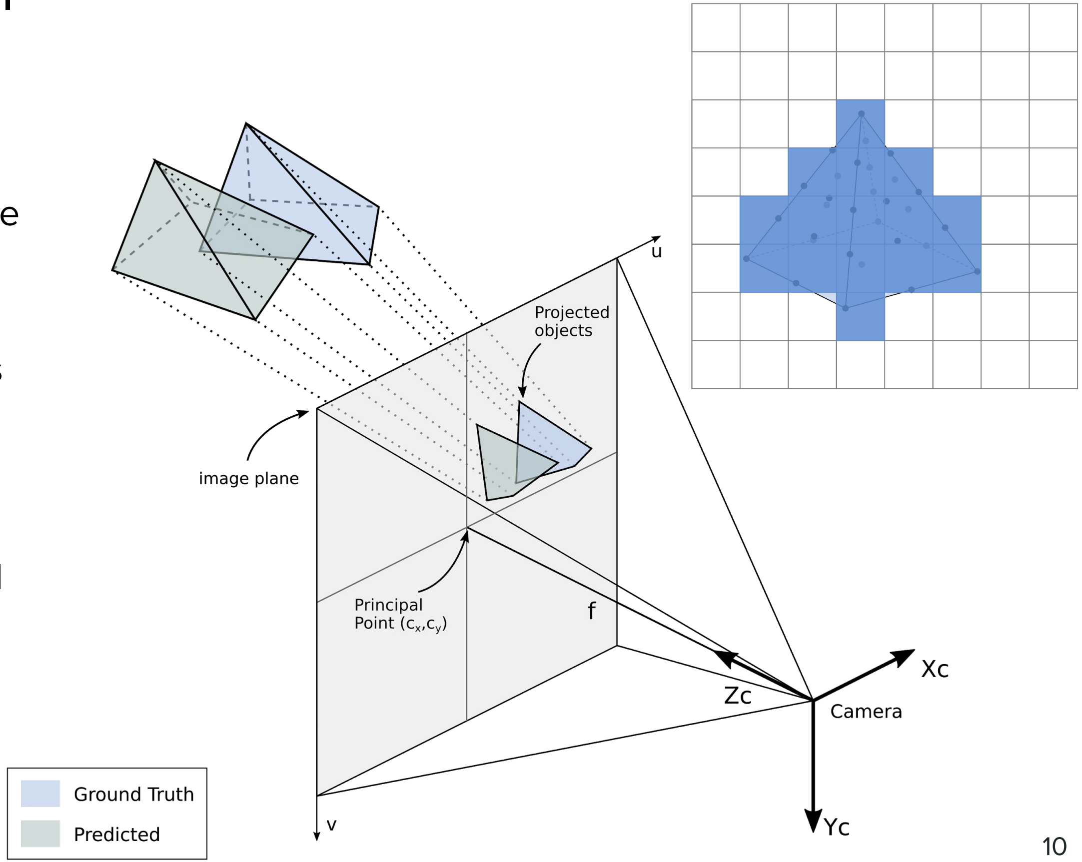
3.2. Projection Loss II

In order to compute the loss using the 3D model point cloud:

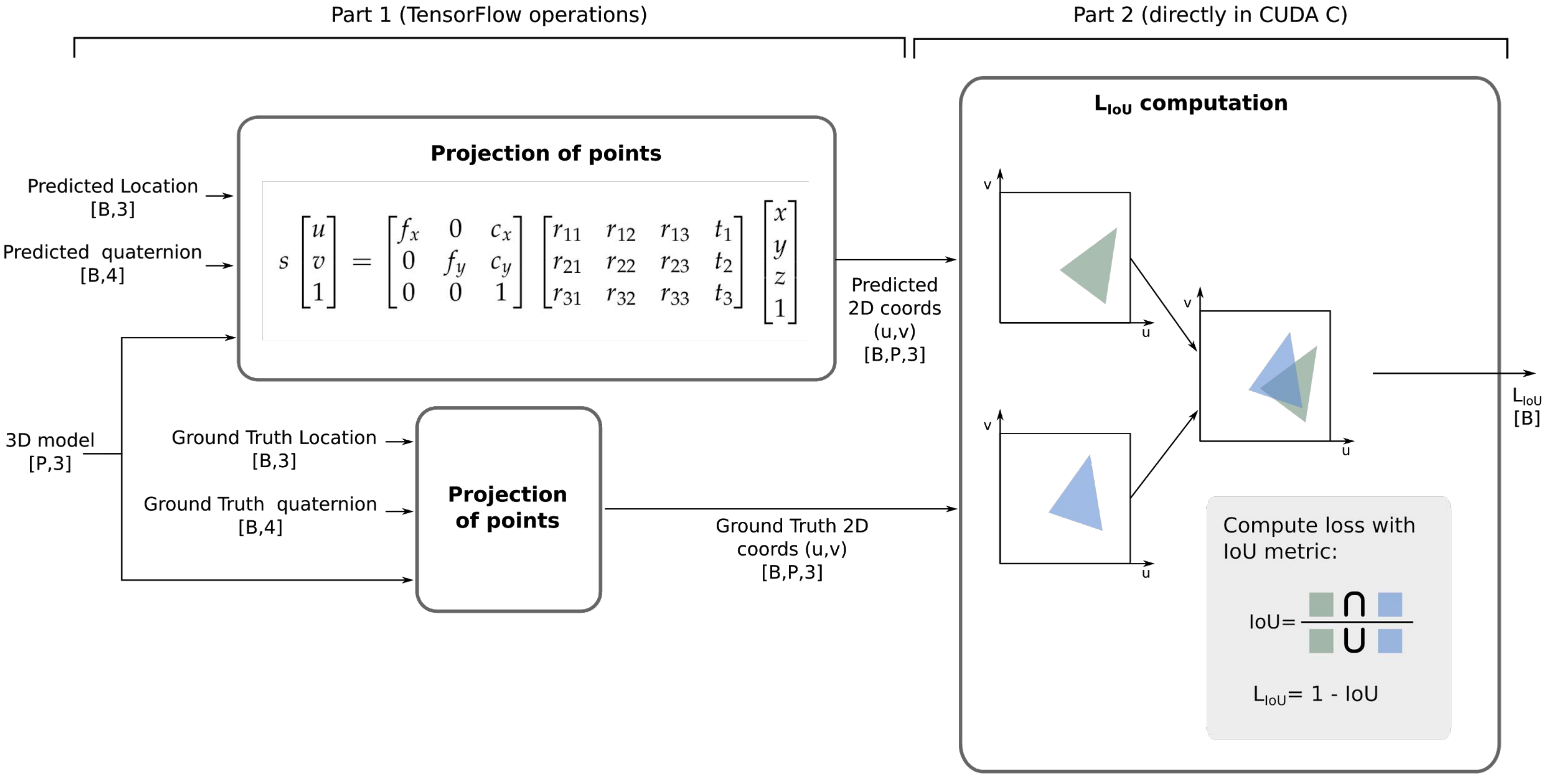
- We compute the projected image of the ground truth and predicted pose with the pinhole camera model.
- We compare the two projected images with the IoU metric.

In practice:

- This operation cannot be implemented entirely in TensorFlow.
- We use CUDA C to compute the projected images and we need to define the forward and the backward pass of the operation



3.2. Projection Loss - Forward pass



3.2. Projection Loss - Backward pass I

We need to find:

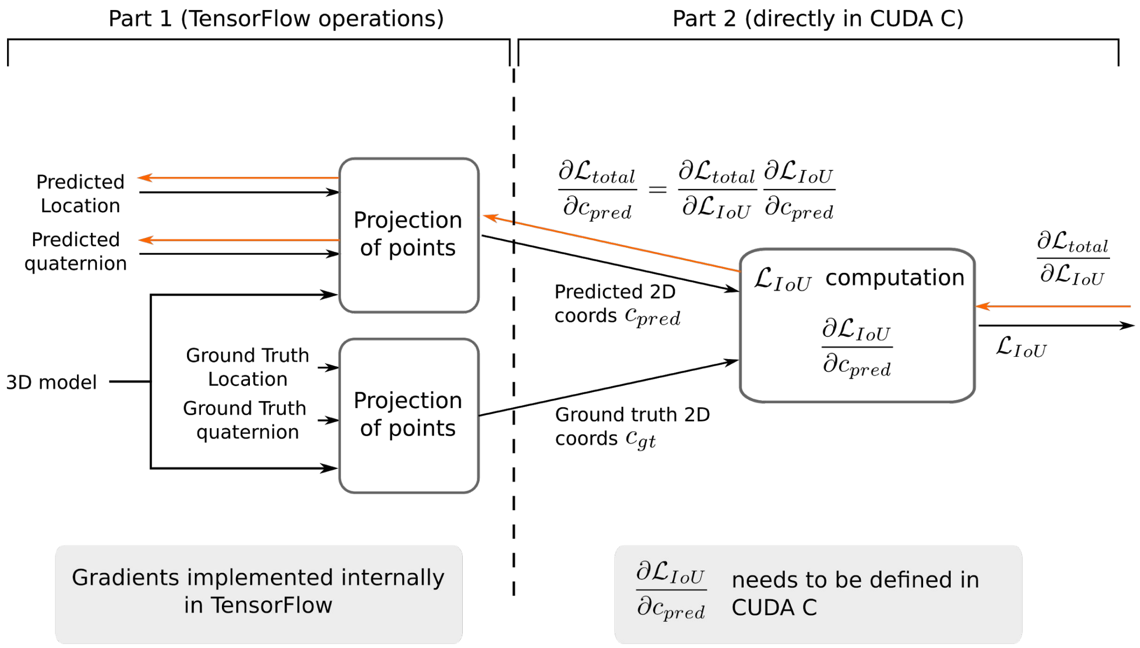
$$\frac{\partial \mathcal{L}_{IoU}}{\partial u_i} = \frac{\partial \mathcal{L}_{IoU}}{\partial X_v} \cdot \frac{\partial X_v}{\partial u_i}$$

The first part of the $\frac{\partial \mathcal{L}_{IoU}}{\partial X_v}$ is simple to derive [1]:

$$\frac{\partial \mathcal{L}_{IoU}}{\partial X_v} = \begin{cases} -\frac{1}{U(X,Y)} & \text{if } Y_v = 1 \\ \frac{I(X,Y)}{U(X,Y)^2} & \text{otherwise} \end{cases}$$

But the sampling operation is a discrete operation, so it is not differentiable:

$$\frac{\partial X_v}{\partial u_i} = ?? \qquad \frac{\partial X_v}{\partial v_i} = ??$$



[1] Md Atiqur Rahman and Yang Wang. “Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation”. In: vol. 10072. Dec. 2016

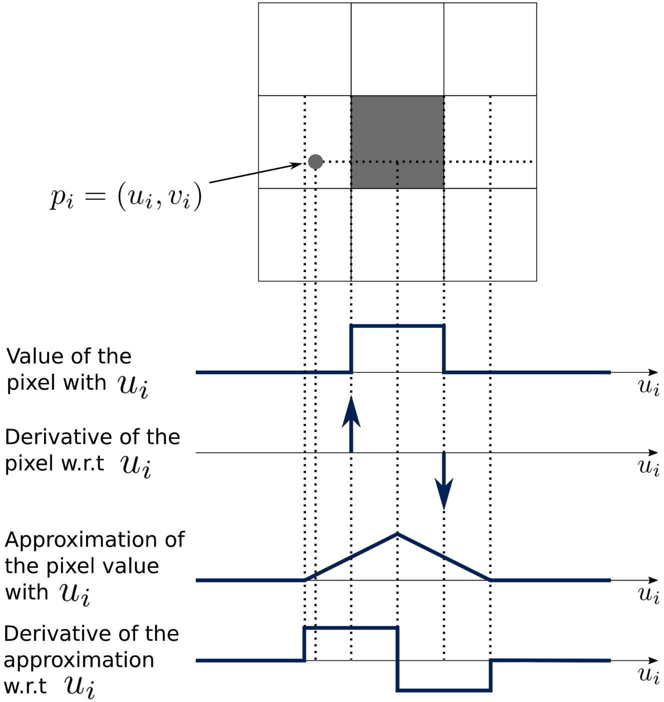
3.2. Projection Loss - Backward pass II

The value of each pixel is approximated to a function that can backpropagate the gradients [2-3].

$$\frac{\partial X_v}{\partial u_i} = \begin{cases} D \cdot max(0, 1 - |n - D \cdot v_i|) & \text{if } D \cdot u_i \leq m \\ -D \cdot max(0, 1 - |n - D \cdot v_i|) & \text{if } D \cdot u_i > m \\ 0 & \text{if } |n - D \cdot u_i| \geq 1 \end{cases}$$

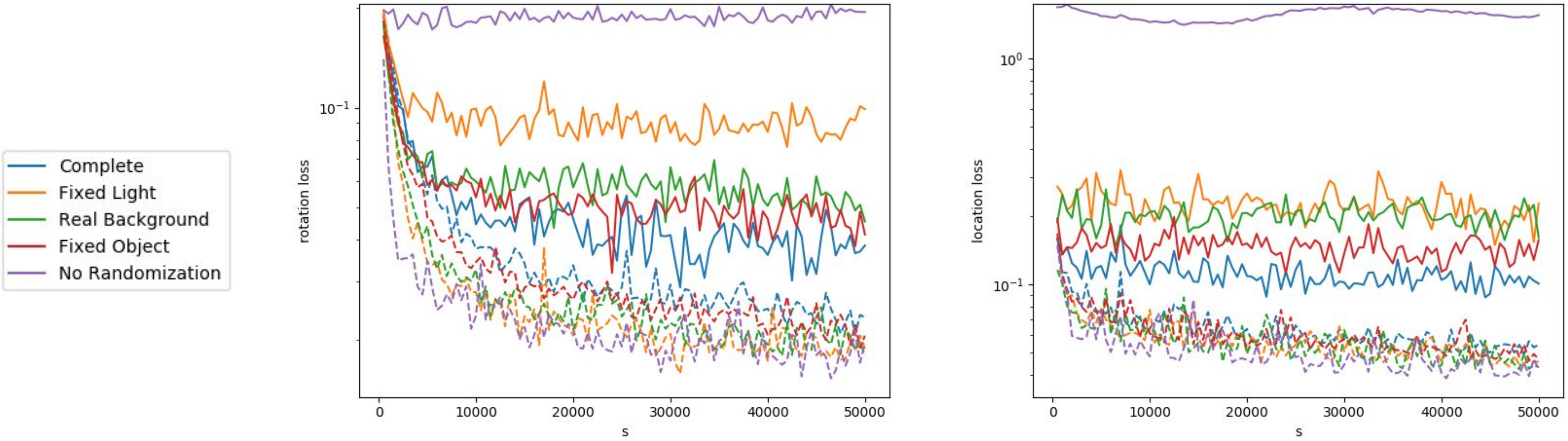
Only propagate gradients:

- One 2D point per pixel in the projected image.
- If the change in the coordinate actually causes a change in the IoU.



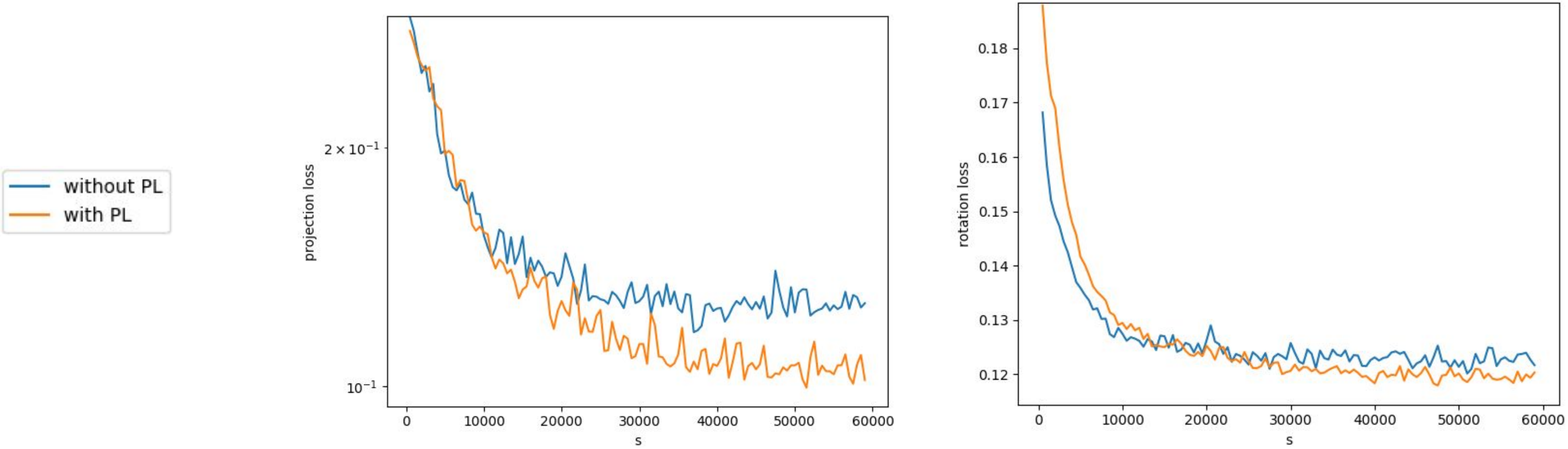
[2] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. “Neural 3D Mesh Renderer”. In: (Nov. 20, 2017)
[3] Max Jaderberg et al. “Spatial Transformer Networks”. In: (June 5, 2015)

4. Results - Domain Randomization



	Synthetic Images		Real Images	
	ADD	ADD-S	ADD	ADD-S
Complete	67.8	71.9	43.9	48.7
Fixed Light	78.4	81.3	21.7	29.6
Real Background	81.3	84.7	24.3	27.0
Fixed Object	72.5	76.9	41.2	44.7
Without Randomization	81.3	84.7	0.0	0.0

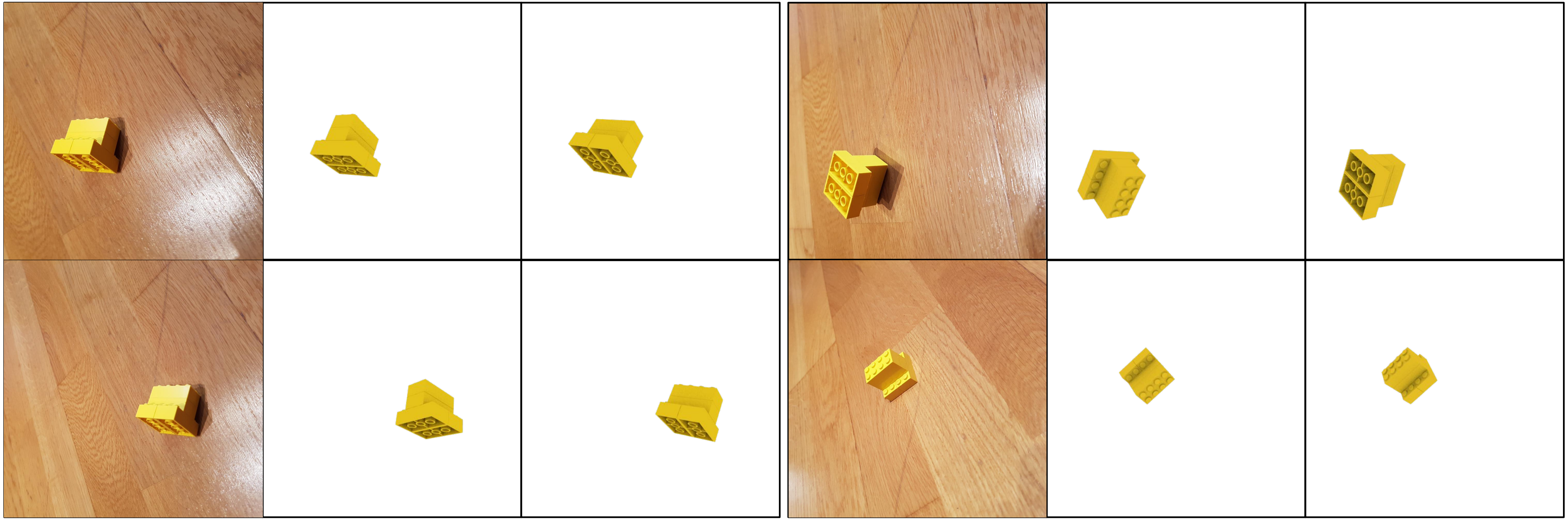
4. Results - Projection Loss I



	ADD	ADD-S
Without Projection Loss	34.5	51
With Projection Loss	36.2	60.7

4. Results - Projection Loss II

Original image | Without PL | With PL | Original image | Without PL | With PL



4. Results - Final I

	ADD	ADD-S
Toy	43.9	48.7
Box	69.0	74.3
Symmetric	36.2	60.7

4. Results - Final II

But still...

5. Conclusions

- The 6D pose regression task was addressed with Deep Models trained exclusively with synthetic data.
- The reality gap problem can be tackled with domain randomization.
- The proposed multi-task Neural Network was effective for the rotation and localization regression.
- The new Projection Loss function was able to deal with ambiguities and symmetric views.

Directions of future work

- Improve the domain randomization
- Use Generative Adversarial Networks to add realism to synthetic images.
- Use a Render Loss instead of a Projection Loss
- Multi-view 6D pose regression

Thank you for your attention

Deep 3D Pose Regression of Real Objects Trained With Synthetic Data

Author:
Pau Bramon Mora

Supervisor:
Sergio Escalera Guerrero