CSC2457 3D & Geometric Deep Learning

NeMo: Neural Mesh Models of Contrastive Features for Robust 3D Pose Estimation

Angtian Wang, Adam Kortylewski, Alan Yuille

Date: March 23rd 2021

Presenter: William Ngo

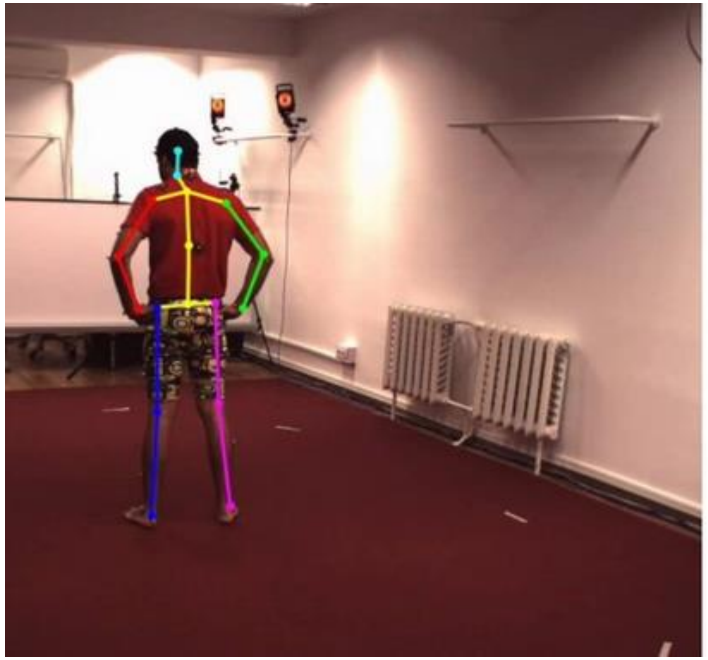
Instructor: Animesh Garg

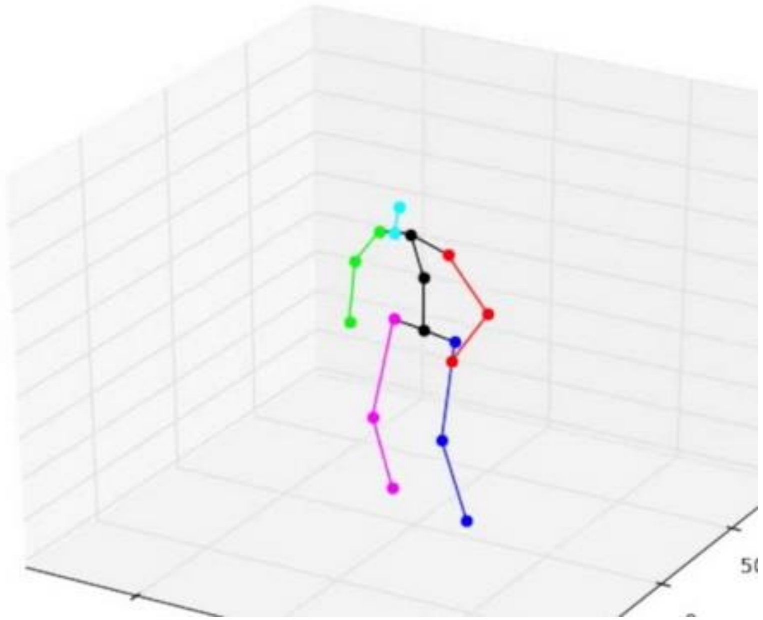
UNIVERSITY OF TORONTO

Main Problem

Goal: (Robust) 3D Pose Estimation

Pose obtained either via its viewpoint or via specifying the locations of a fixed set of keypoints

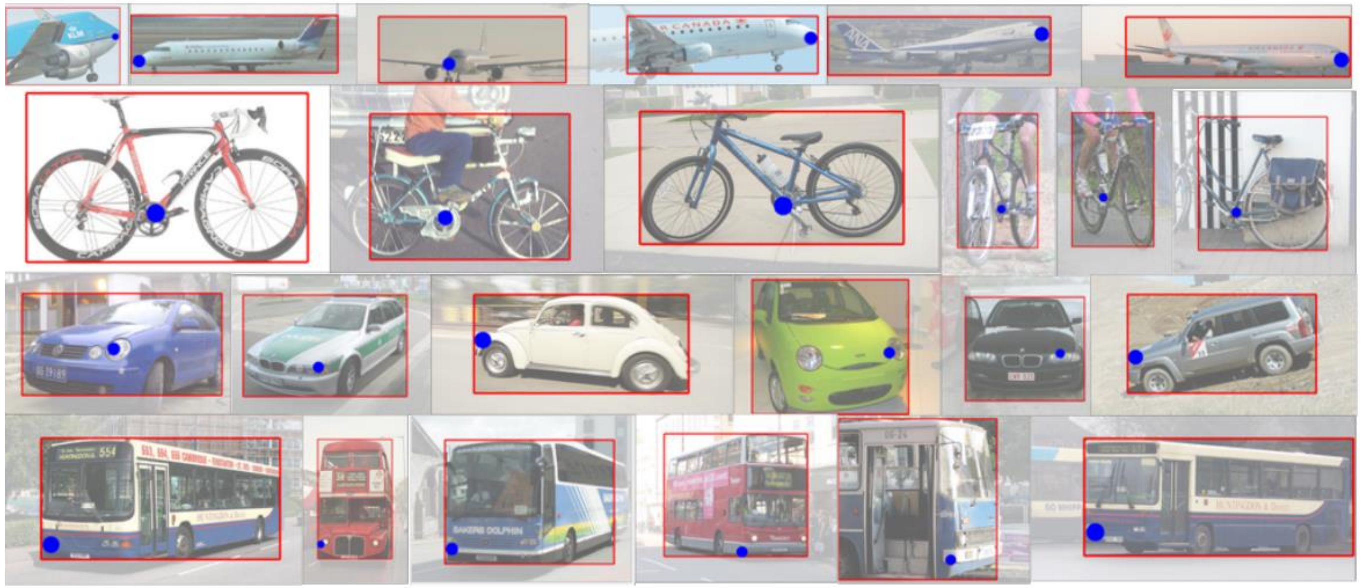




Main Problem

Previous Method:

1. Keypoint-based approaches
   - Detect sparse set of key points & align a 3D object representation to the detection result.



(Tulsiani et. al, 2015)

Main Problem

- Desire: robustness to occlusion

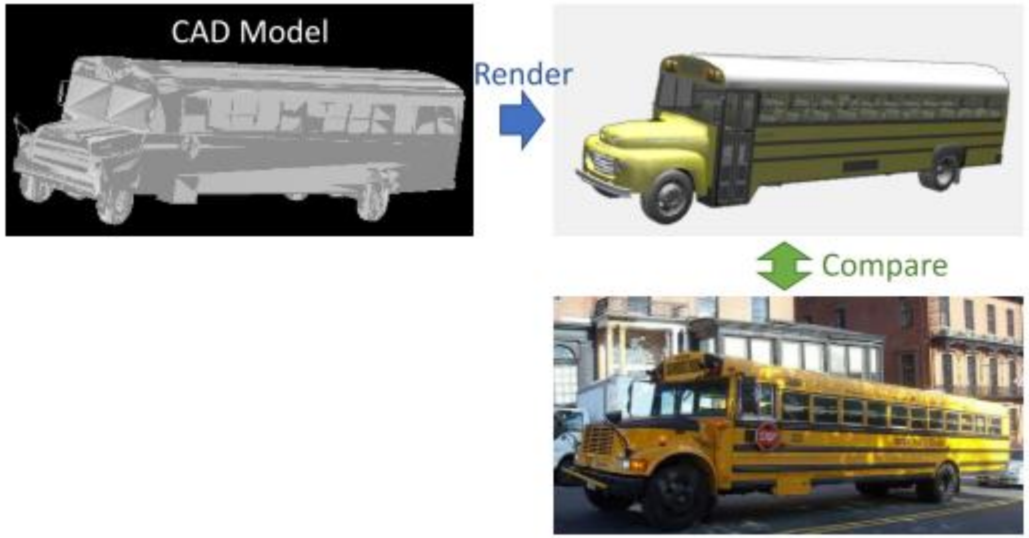
(Zia et. al, 2013)

Main Problem

Previous Method

2. Rendering-based approaches

- Utilize a generative model, that is built on a dense 3D mesh representation of an object. They estimate the object pose by reconstructing the input image (render-and-compare)



Problem:

- They model objects in terms of image intensities
- Color is not relevant to pose estimation!
- Mesh Representation for every shape instance

Contributions

- Develop Framework for 3D Pose Estimation Under Occlusion
  - Generative Model of Features in terms of the mesh input
  - Previous rendering-based approaches require detailed instance-specific mesh representations of targets
    - NeMo achieves competitive 3D pose estimation using a mesh representation which only crudely approximates the true object geometry with a cuboid
  - State of the art performance on PASCAL3D+, occluded-PASCAL3D+ and ObjectNet3D

General Background

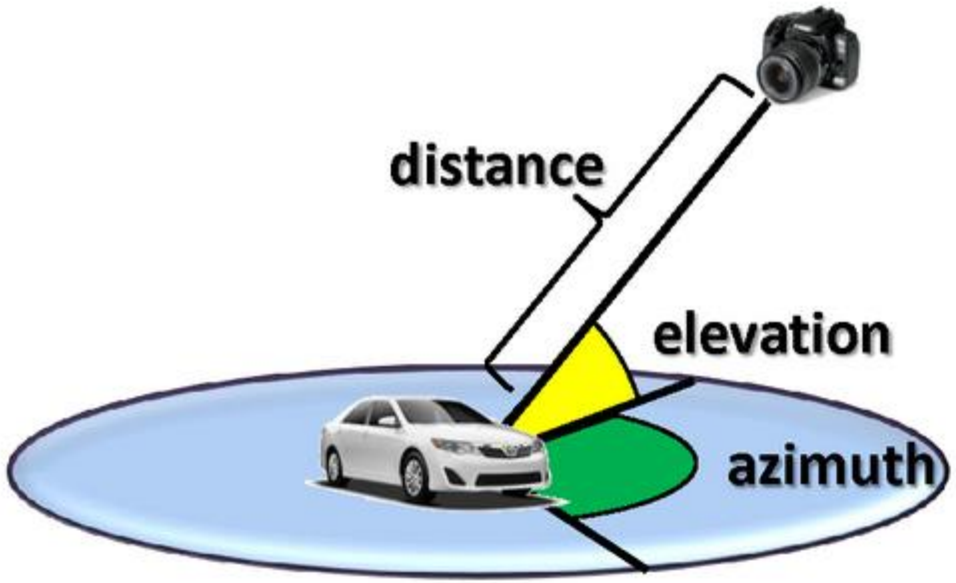
3D object pose estimation involves prediction of 3 spherical angles:

- Azimuth (a)
- Elevation (e)
- In-plane rotation (θ)

Of an object relative to the camera

Define a Rotation Matrix

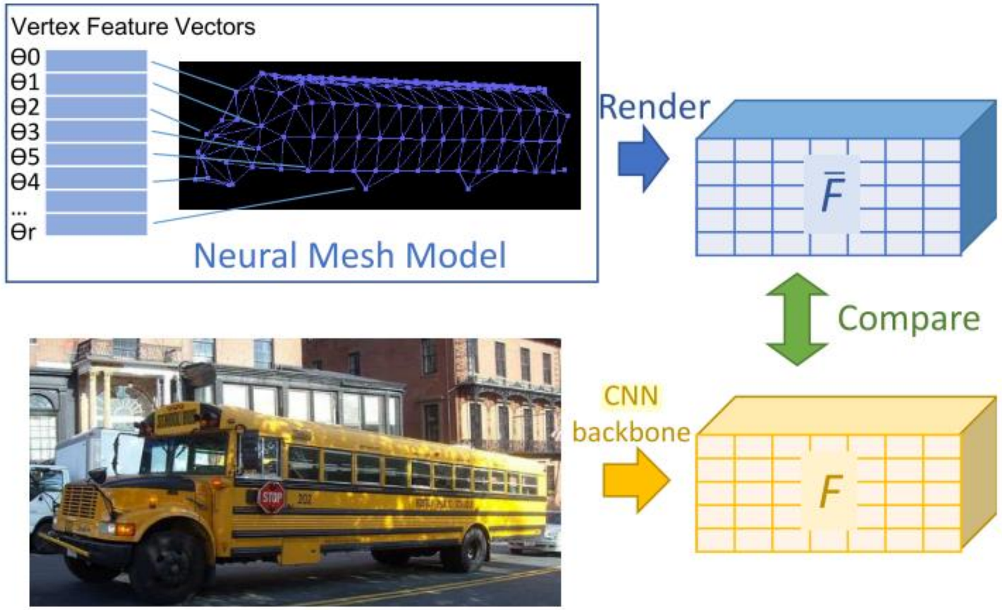
$$R = R_Z(\theta) R_X(e - \pi/2) R_Z(-a)$$



(Xiang et. al, 2014)

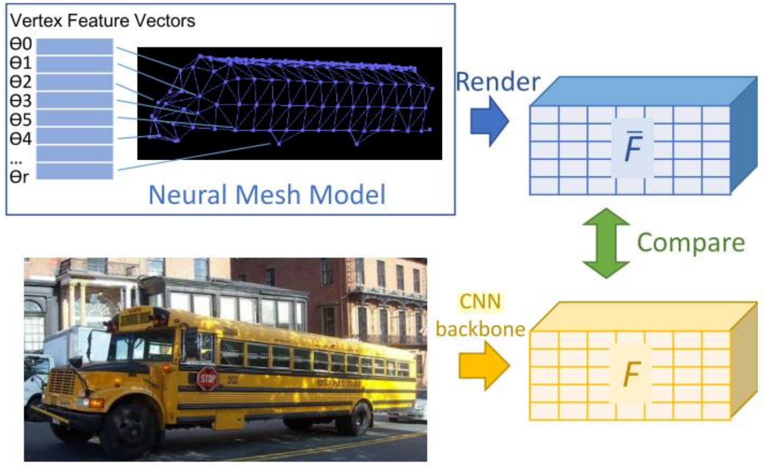
Problem Setting

Goal: Determine rotation matrix with respect to the input image, given target class and the given mesh of the object.



NeMo Render-and-Compare: Feature Map

Problem Setting



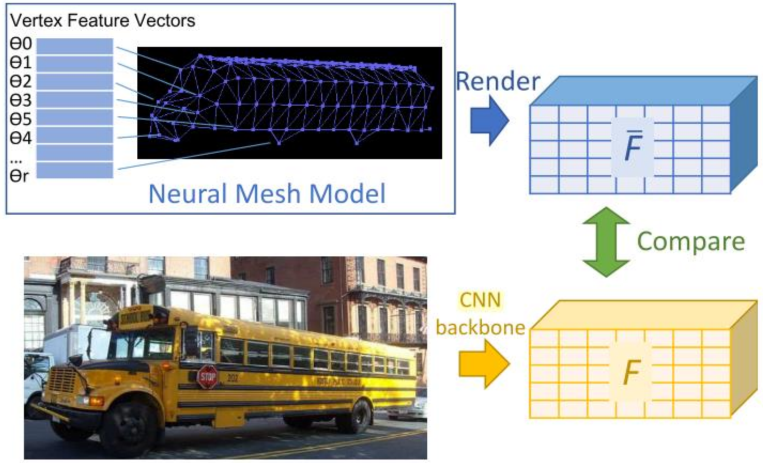
NeMo Render-and-Compare: Feature Map

Let's denote the following:

- Feature representation of the input image I: $\Phi(I) = F^l \in \mathbb{R}^{H \times W \times D}$
  - l Denotes the output of a layer l of a CNN

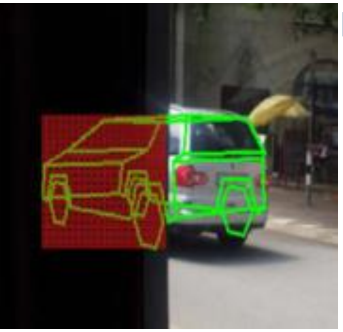
- 3D vertices of mesh: $\Gamma = \{r \in \mathbb{R}^3 | r = 1, \ldots, R\}$
- Feature Vectors at each vertex: $\Theta = \{\theta_r \in \mathbb{R}^D | r = 1, \ldots, R\}$
- 3D Neural Mesh Model: $\mathfrak{N} = \{\Gamma, \Theta\}$
- Rendered Feature Map: $\bar{F}(m) = \Re(\mathfrak{N}, m) \in \mathbb{R}^{H \times W \times D}$
  - m: camera pose (ground truth rotation is used during training time)

Approach



NeMo Render-and-Compare: Feature Map

- Learn Generative Model $\bar{F} : p(F|\mathfrak{N}_y)$
  - True Distribution should follow the feature extracted from the CNN backbone (F)
- Define Likelihood as follows: $p(F|\mathfrak{N}_y, m, B) = \prod_{i \in \mathcal{FG}} p(f_i|\mathfrak{N}_y, m) \prod_{i' \in \mathcal{BG}} p(f_{i'}|B).$
  - $\mathcal{FG}$ is set of all positions on the 2D lattice of the feature map F that are **covered by the neural mesh mode**



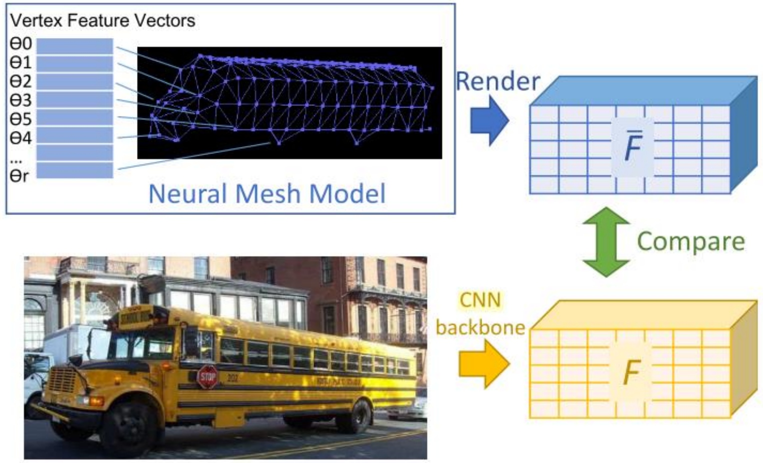
    - Calculated by projecting mesh onto the image using the ground truth camera pose m
    - Think of it as **visible projected vertices in the image**

- 3D vertices of mesh: $\Gamma = \{r \in \mathbb{R}^3 | r = 1, \ldots, R\}$
- Feature Vectors at each vertex: $\Theta = \{\theta_r \in \mathbb{R}^D | r = 1, \ldots, R\}$
- 3D Neural Mesh Model: $\mathfrak{N} = \{\Gamma, \Theta\}$

Approach

- Define foreground feature likelihood to be Gaussian: $p(f_i|\mathfrak{N}_y, m) = \frac{1}{\sigma_r\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma_r^2}\|f_i - \theta_r\|^2\right)$
  - Note: the correspondence between θ_r and f_i is defined between the projection of the vertices onto the 2D lattice given the parameter camera pose.
- Background features are also modelled as Gaussian: $p(f_{i'}|B) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\|f_{i'} - \beta\|^2\right)$
  - Mean background Vector, “clutter vector”: β

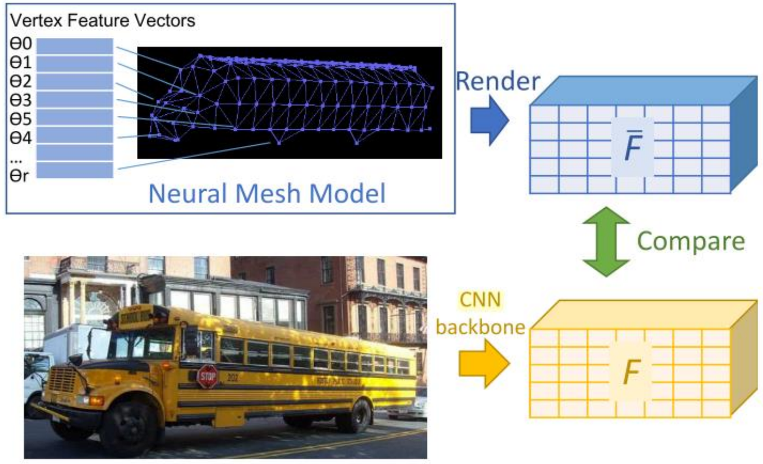
Approach



NeMo Render-and-Compare: Feature Map

- Want to optimize the following:
  - Maximum likelihood to such that the generative model's distribution matches with the image features (Make $\bar{F}$ as close as possible to F)
  - The CNN backbone used for feature extraction should be optimized to make the individual feature vectors as distinct from each other as possible (Make features in F as distinct as possible)

Approach



NeMo Render-and-Compare: Feature Map

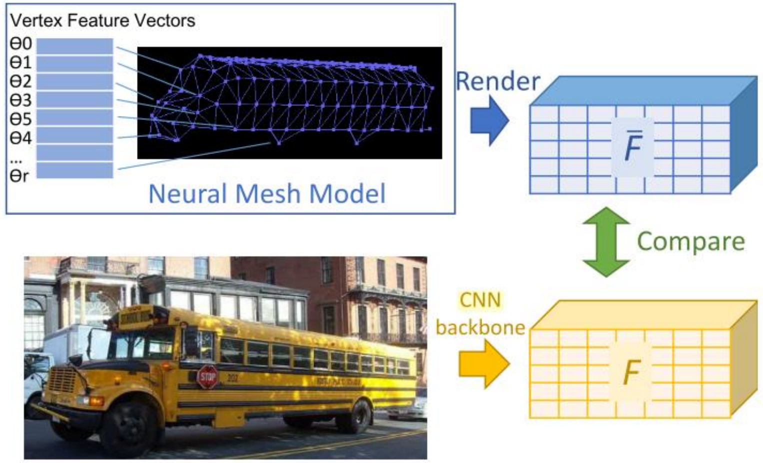
Maximum Likelihood Estimation of the generative Model:

$$\begin{aligned}\mathcal{L}_{ML}(F, \mathfrak{N}_y, m, B) &= -\ln p(F|\mathfrak{N}_y, m, B) \\ &= -\sum_{i \in \mathcal{FG}} \ln\left(\frac{1}{\sigma_r\sqrt{2\pi}}\right) - \frac{1}{2\sigma_r^2}\|f_i - \theta_r\|^2 \\ &\quad + \sum_{i' \in \mathcal{BG}} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2\sigma^2}\|f_{i'} - \beta\|^2\end{aligned}$$

If we constrain the variances: $\{\sigma^2 = \sigma_r^2 = 1|\forall r\}$

$$\mathcal{L}_{ML}(F, \mathfrak{N}_y, m, B) = -C \sum_{i \in \mathcal{FG}} \|f_i - \theta_r\|^2 + \sum_{i' \in \mathcal{BG}} \|f_{i'} - \beta\|^2$$

Approach



NeMo Render-and-Compare: Feature Map

Contrastive Learning in backbone:

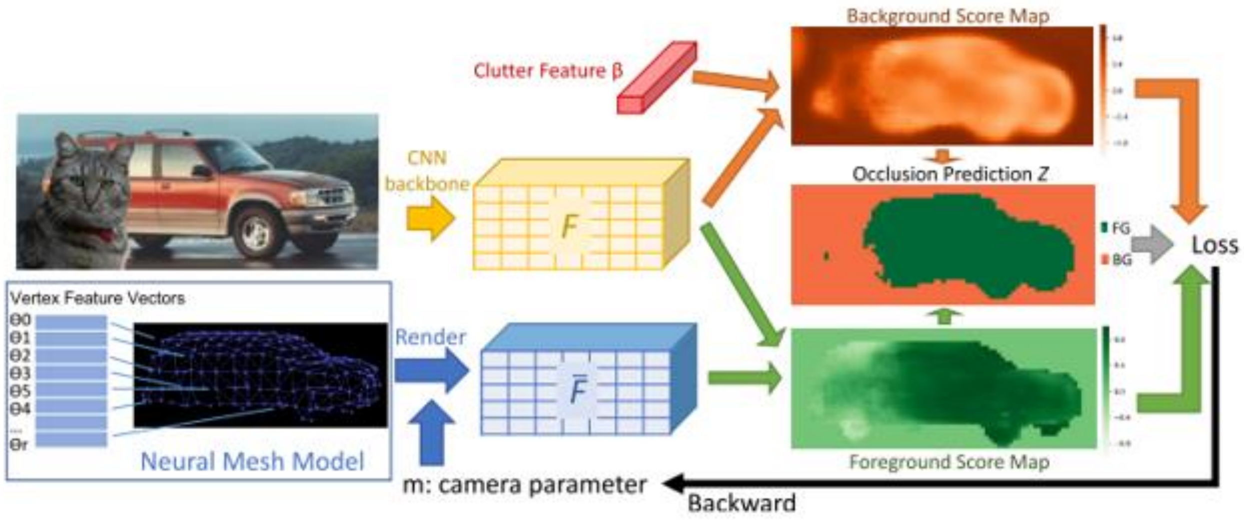
$$\mathcal{L}_{Feature}(F, \mathcal{FG}) = -\sum_{i \in \mathcal{FG}} \sum_{i' \in \mathcal{FG} \setminus \{i\}} \|f_i - f_{i'}\|^2$$

$$\mathcal{L}_{Back}(F, \mathcal{FG}, \mathcal{BG}) = -\sum_{i \in \mathcal{FG}} \sum_{j \in \mathcal{BG}} \|f_i - f_j\|^2.$$

Contrastive Loss encourages features on the object to be distinct from each other (feature vector at front tire should be different from those at the back tire)

Overall Loss: $\mathcal{L}(F, \mathfrak{N}_y, m, B) = \mathcal{L}_{ML}(F, \mathfrak{N}_y, m, B) + \mathcal{L}_{Feature}(F, \mathcal{FG}) + \mathcal{L}_{Back}(F, \mathcal{FG}, \mathcal{BG})$

Approach

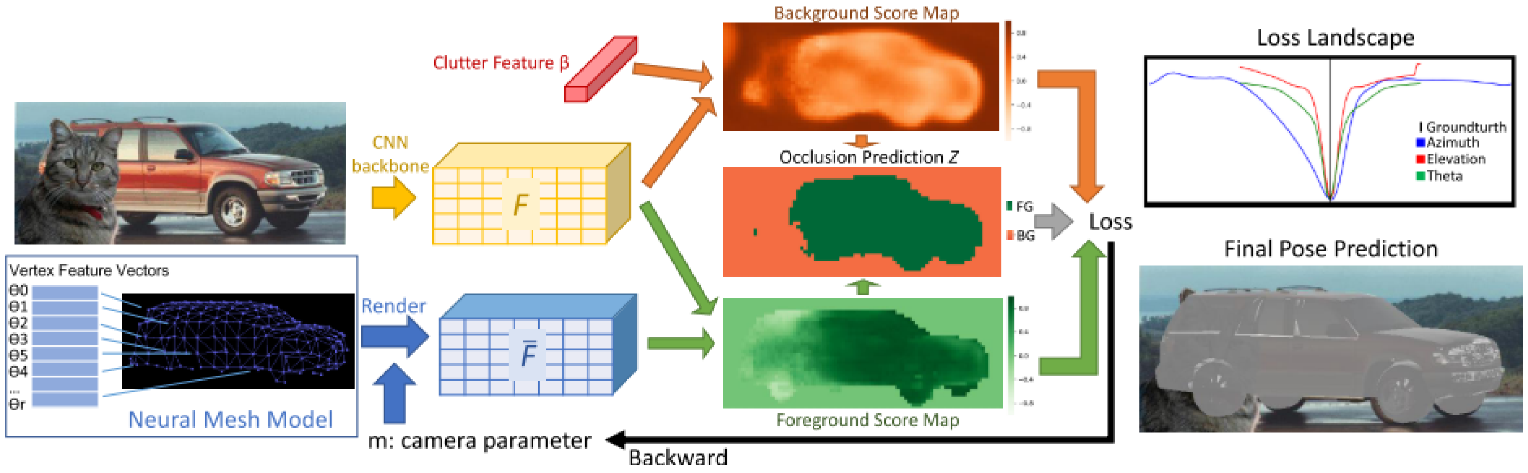


Now we have trained our network, both the CNN backbone (F) and the generative model ($\bar{F}$)

Question: I still don't get it, how does the model determine the camera pose parameter at inference time?

Answer: At inference time, Given an initial camera pose estimate, it will perform gradient descent to find an optimal camera pose estimate.

Approach



Question: With respect to what cost will it perform gradient descent on?

Answer: Reconstruction Loss between the Foreground Score Map and Background Score Map and Reconstruction Loss of (F with $\bar{F}$)

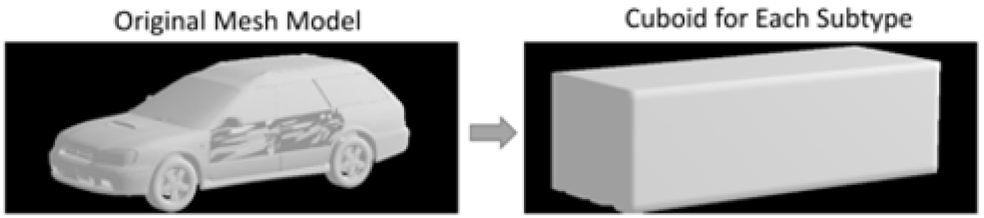
$$p(F|\mathfrak{N}_y, m, B, z_i) = \prod_{i \in \mathcal{FG}} [p(f_i|\mathfrak{N}_y, m)p(z_i{=}1)]^{z_i} [p(f_i|B)p(z_i{=}0)]^{(1-z_i)} \prod_{i' \in \mathcal{BG}} p(f_{i'}|B)$$

- Where z_i is a binary variable that allows the background model $p(f_i|B)$ to explain the locations in F that are in the FG, but the foreground model $(f_i|\mathfrak{N}_y, m)$ can't explain well.

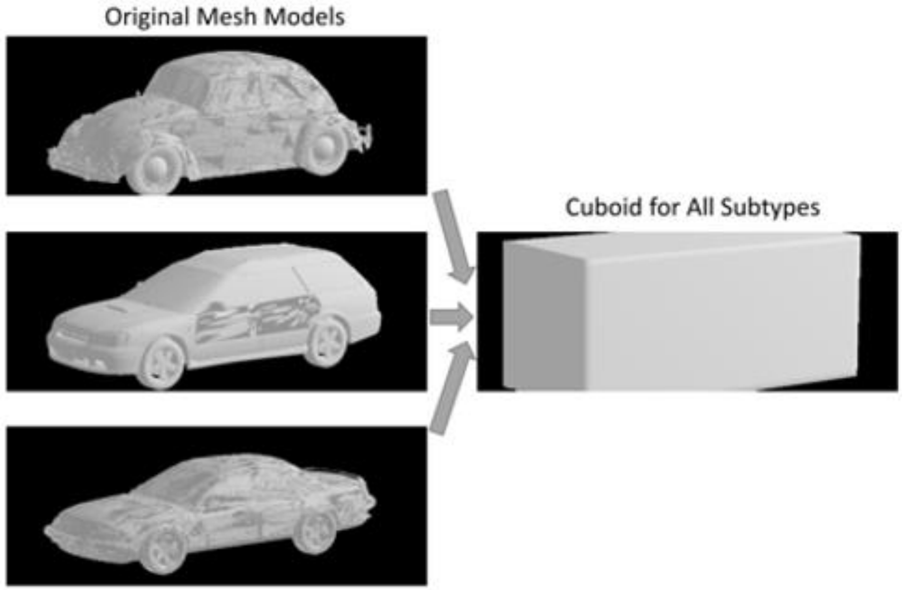
Approach

Question: Wait a second? Don't we need a detailed mesh for every instance?

Answer: No, we can have a much simpler mesh for every instance (cuboid).



(b) MultiCuboid



(c) SingleCuboid

Discussion of results

To evaluate, first define $\Delta(R_1, R_2) = \frac{\|log(R_1^T R_2)\|_F}{\sqrt{2}}$

- It represents the geodesic distance function over the manifold of rotation matrices

$\Delta(R_{gt}, R_{pred})$ captures the difference between ground truth rotation and predicted rotation matrix

They report the following:

- Median of the rotation error
- Accuracy at theta: fraction of instances whose predicted rotation is within a fixed threshold of the target rotation (they use pi/6 and pi/18)

Discussion of results

PASCAL3D+ and Occluded PASCAL3D+ results

Evaluation Metric	$ACC_{\frac{\pi}{6}} \uparrow$				$ACC_{\frac{\pi}{18}} \uparrow$				$MedErr \downarrow$			
Occlusion Level	L0	L1	L2	L3	L0	L1	L2	L3	L0	L1	L2	L3
Res50-General	88.1	70.4	52.8	37.8	44.6	25.3	14.5	6.7	11.7	17.9	30.4	46.4
Res50-Specific	87.6	73.2	58.4	43.1	43.9	28.1	18.6	9.9	11.8	17.3	26.1	44.0
StarMap	**89.4**	71.1	47.2	22.9	59.5	34.4	13.9	3.7	9.0	17.6	34.1	63.0
NeMo	84.1	73.1	59.9	41.3	60.4	45.1	30.2	14.5	9.3	15.6	24.1	41.8
NeMo-MultiCuboid	86.7	**77.2**	**65.2**	**47.1**	**63.2**	**49.9**	**34.5**	**17.8**	**8.2**	**13.0**	**20.2**	**36.1**
NeMo-SingleCuboid	86.1	76.0	63.9	46.8	61.0	46.3	32.0	17.1	8.8	13.6	20.9	36.5

Discussion of results

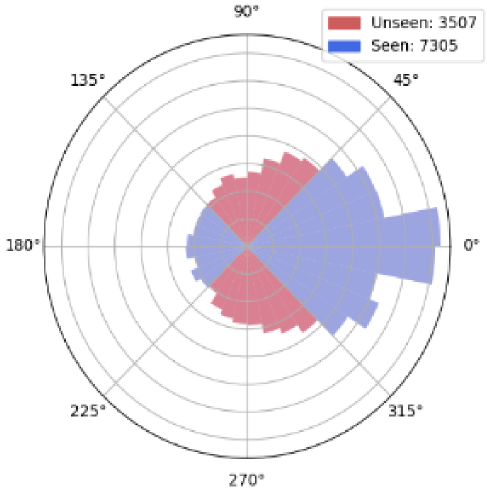
ObjectNet3D results

$ACC_{\frac{\pi}{6}} \uparrow$	bed	bookshelf	calculator	cellphone	computer	cabinet	guitar	iron	knife
StarMap	40.0	**72.9**	21.1	**41.9**	62.1	79.9	38.7	2.0	6.1
NeMo-MultiCuboid	**56.1**	53.7	**57.1**	28.2	**78.8**	**83.6**	**38.8**	**32.3**	**9.8**
$ACC_{\frac{\pi}{6}} \uparrow$	microwave	pen	pot	rifle	slipper	stove	toilet	tub	wheelchair
StarMap	86.9	**12.4**	45.1	3.0	**13.3**	79.7	35.6	46.4	17.7
NeMo-MultiCuboid	**90.3**	3.7	**66.7**	**13.7**	6.1	**85.2**	**74.5**	**61.6**	**71.7**

ObjectNet3D is more occluded than occluded PASCAL3D++

Discussion of results

Generalization to unseen views



Evaluation Metric	$ACC_{\frac{\pi}{6}} \uparrow$		$ACC_{\frac{\pi}{18}} \uparrow$		$MedErr \downarrow$	
Data Split	Seen	Unseen	Seen	Unseen	Seen	Unseen
Res50-General	91.7	37.2	47.9	5.3	10.8	45.8
Res50-Specific	91.2	34.7	47.9	4.0	10.8	48.5
StarMap	**93.1**	49.8	68.6	13.5	7.3	36.0
NeMo-MultiCuboid	88.6	**54.7**	**70.2**	**31.0**	**6.6**	**34.9**
NeMo-SingleCuboid	88.5	54.3	68.6	27.9	7.0	35.1

Discussion of results

Ablation Study

Table 4: Ablation study on PASCAL3D+ and occluded PASCAL3D+. All ablation experiments are conducted with the NeMo-MultiCuboid model. The performance is reported in terms of Accuracy (percentage, higher better) and Median Error (degree, lower better).

Evaluation Metric	$ACC_{\frac{\pi}{6}} \uparrow$				$ACC_{\frac{\pi}{18}} \uparrow$				$MedErr \downarrow$			
Occlusion Level	L0	L1	L2	L3	L0	L1	L2	L3	L0	L1	L2	L3
NeMo	**86.7**	**77.3**	**65.2**	**47.1**	**63.2**	**49.2**	**34.5**	**17.8**	**8.2**	**13.1**	**20.2**	**36.1**
NeMo w/o outlier	85.2	76.0	63.2	44.4	61.8	47.9	32.4	16.2	8.5	13.5	20.7	41.6
NeMo w/o contrastive	69.7	58.0	44.6	26.9	40.8	27.7	14.7	5.6	18.3	27.7	37.0	61.0

Critique / Limitations / Open Issues

- Doing gradient descent at inference time is expensive!
  - 8 seconds per image on a single GPU.
- Still requires a cuboid mesh matching with a similar dimension as the object with minimum volume
  - Consider trying cuboid meshes with larger volume than necessary
- Neural Mesh Model for each subtype in a category is trained

Contributions (recap)

- Developped Framework for 3D Pose Estimation Under Occlusion
  - Generative Model of Features in terms of the mesh input
  - Previous rendering-based approaches require detailed instance-specific mesh representations of targets
    - NeMo achieves competitive 3D pose estimation using a mesh representation which only crudely approximates the true object geometry with a cuboid
  - State of the art performance on PASCAL3D+, occluded-PASCAL3D+ and ObjectNet3D