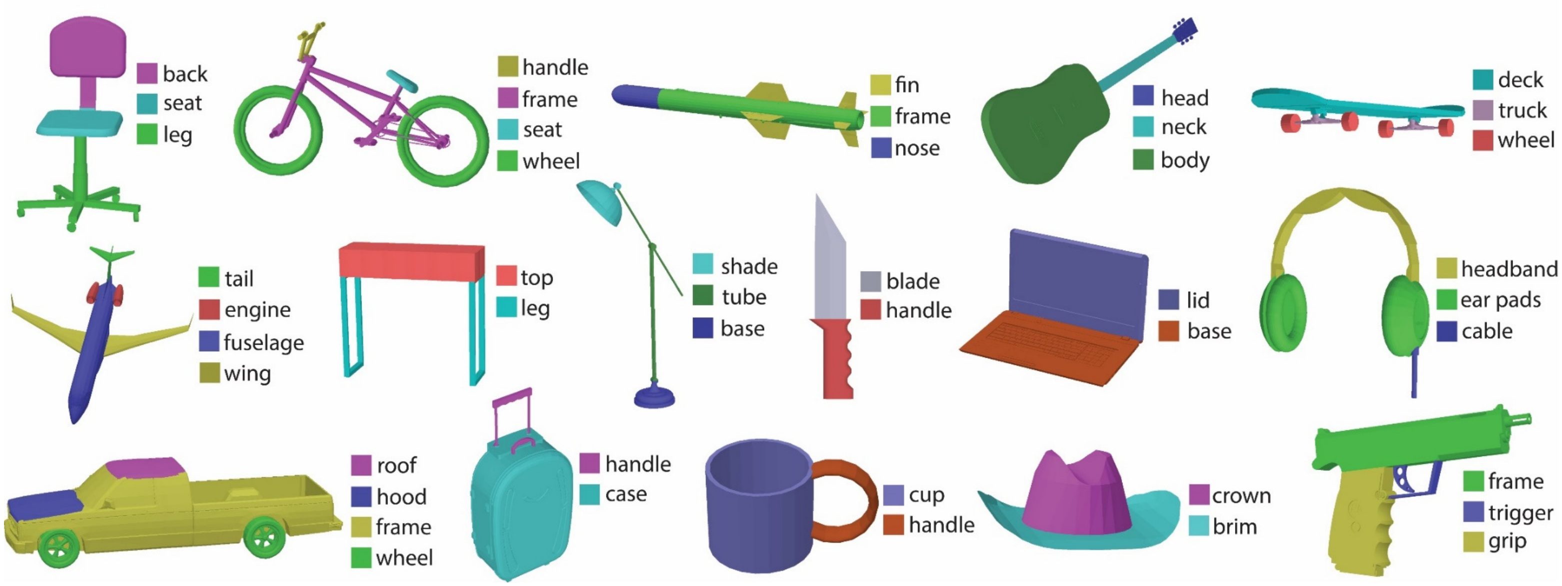
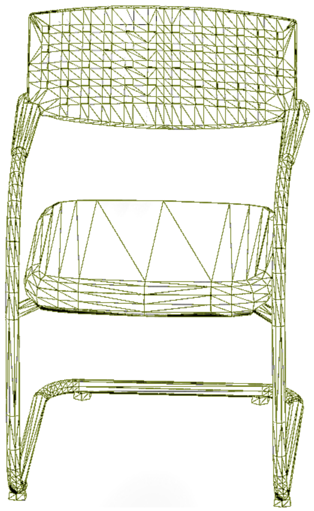
3D Shape Segmentation with Projective Convolutional Networks



Evangelos Kalogerakis[1] Melinos Averkiou[2] Subhransu Maji[1] Siddhartha Chaudhuri[3]

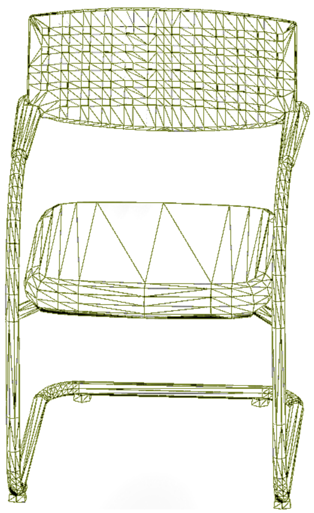
[1]University of Massachusetts Amherst [2]University of Cyprus [3]IIT Bombay

Goal: learn to segment & label parts in 3D shapes

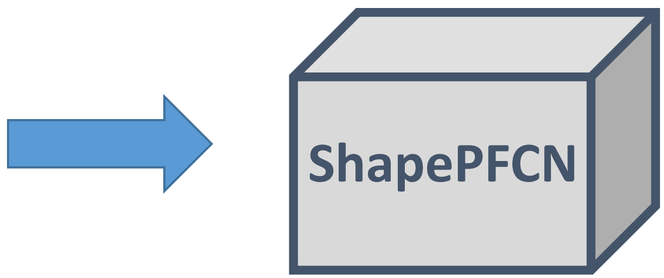


3D Geometric representation

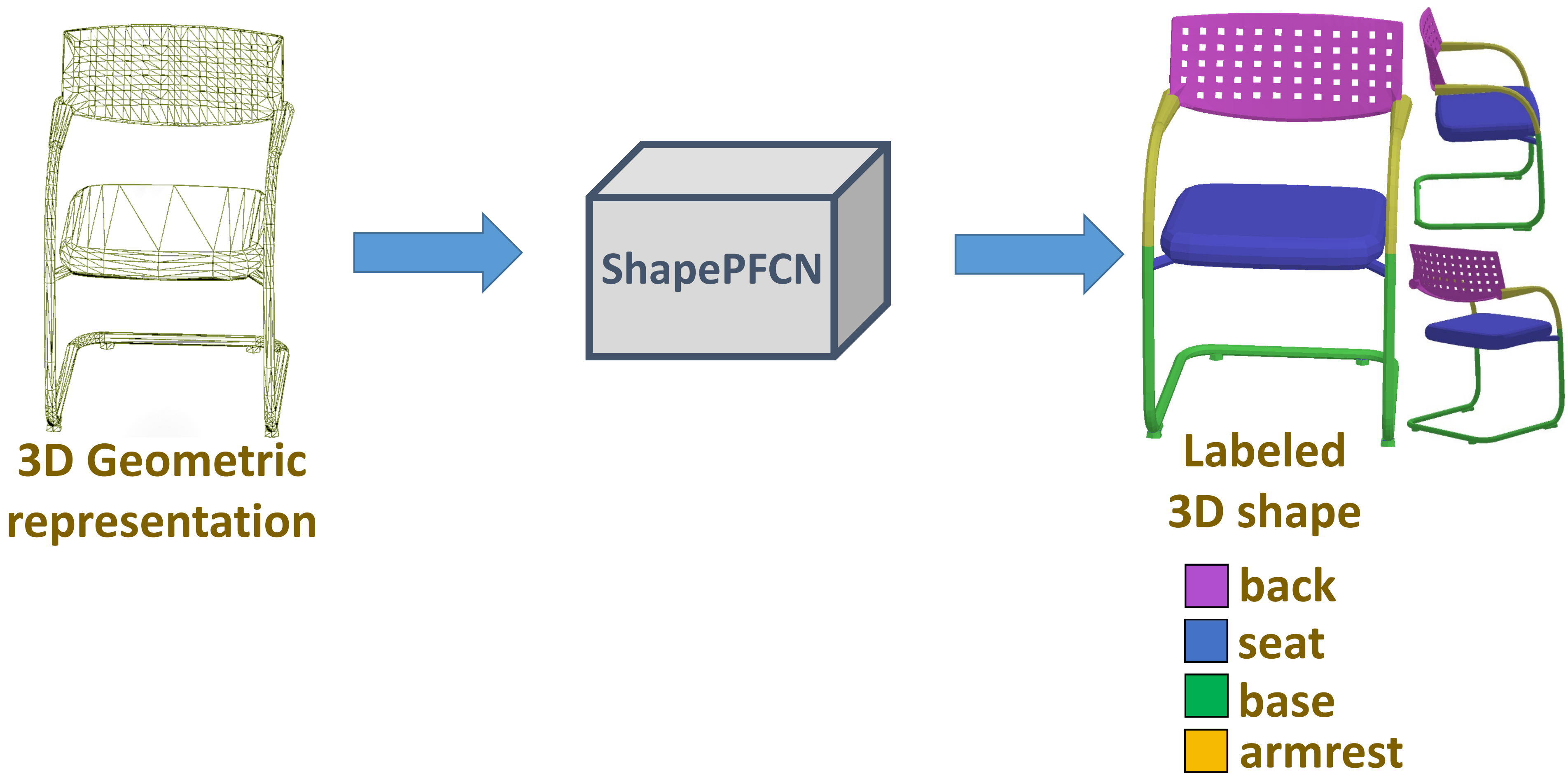
Goal: learn to segment & label parts in 3D shapes



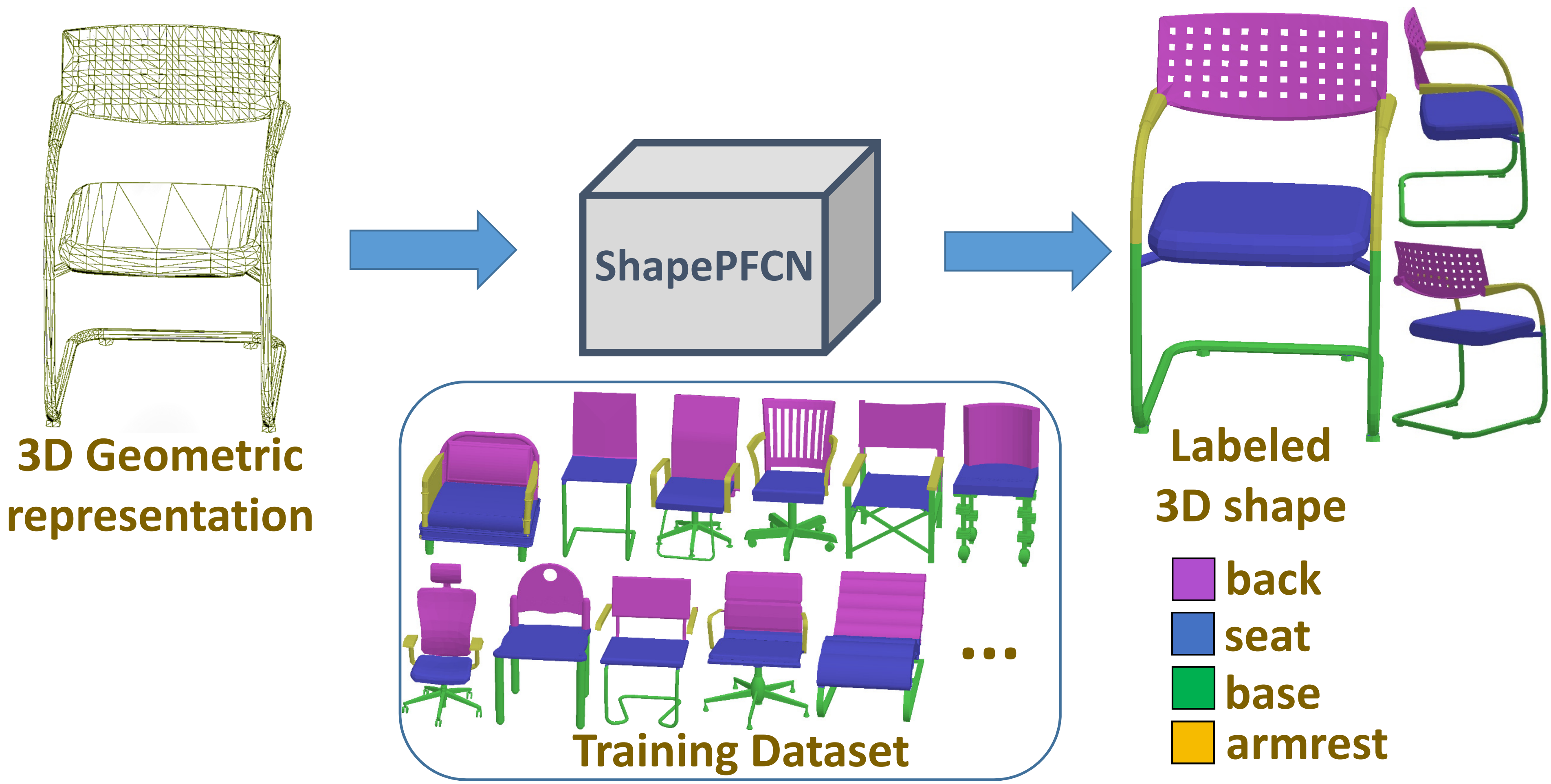
3D Geometric representation



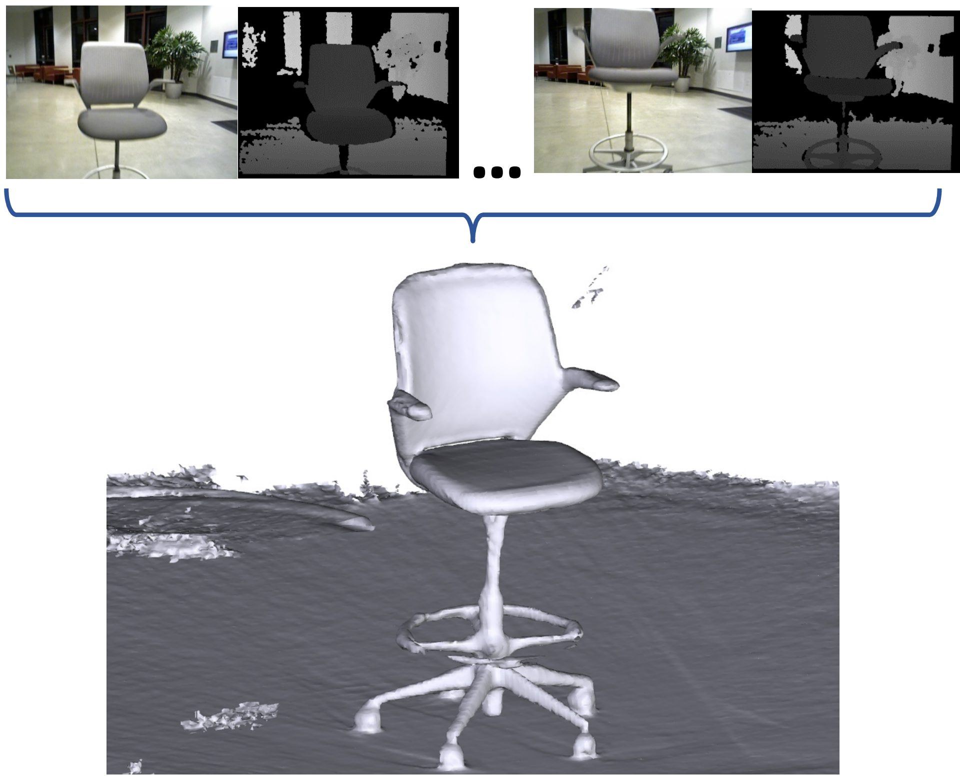
Goal: learn to segment & label parts in 3D shapes



Goal: learn to segment & label parts in 3D shapes



Motivation: Parsing RGBD data

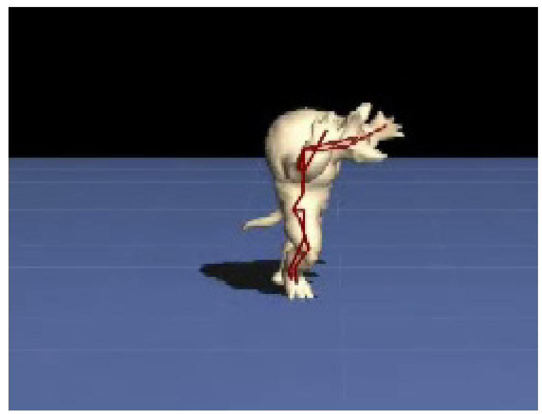


RGBD data from
"A Large Dataset of Object Scans"
Choi, Zhou, Miller, Koltun 2016

Our result

Motivation: 3D Modeling & Animation

3D Models



Animation

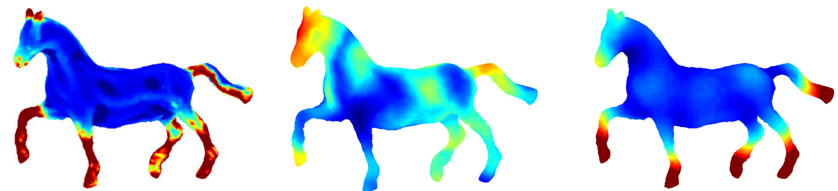
Texturing

Kalogerakis, Hertzmann, Singh, SIGGRAPH 2010

Simari, Nowrouzezahrai, Kalogerakis, Singh, SGP 2009

Related Work

Train classifiers on hand-engineered descriptors
e.g., **Kalogerakis et al. 2010, Guo et al. 2015**



Related Work

Train classifiers on hand-engineered descriptors
e.g., **Kalogerakis et al. 2010, Guo et al. 2015**

Concurrent approaches:

Volumetric / octree-based methods: **Riegler et al. 2017 (OctNet), Wang et al. 2017 (O-CNN), Klokov et al. 2017 (kd-net)**

Point-based networks: **Qi et al. 2017 (PointNet / PointNet++)**

Graph-based / spectral networks: **Yi et al. 2017 (SyncSpecCNN)**

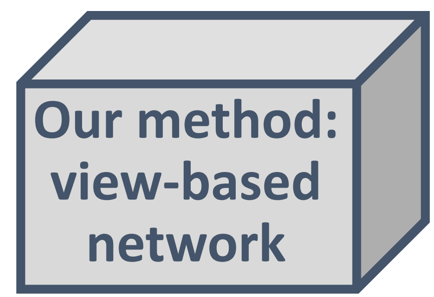
Surface embedding networks: **Maron et al. 2017**

Related Work

Train classifiers on hand-engineered descriptors
e.g., **Kalogerakis et al. 2010, Guo et al. 2015**

Concurrent approaches:

Volumetric / octree-based methods: **Riegler et al. 2017 (OctNet), Wang et al. 2017 (O-CNN), Klokov et al. 2017 (kd-net)**



Point-based networks: **Qi et al. 2017 (PointNet / PointNet++)**

Graph-based / spectral networks: **Yi et al. 2017 (SyncSpecCNN)**

Surface embedding networks: **Maron et al. 2017**

Key Observations

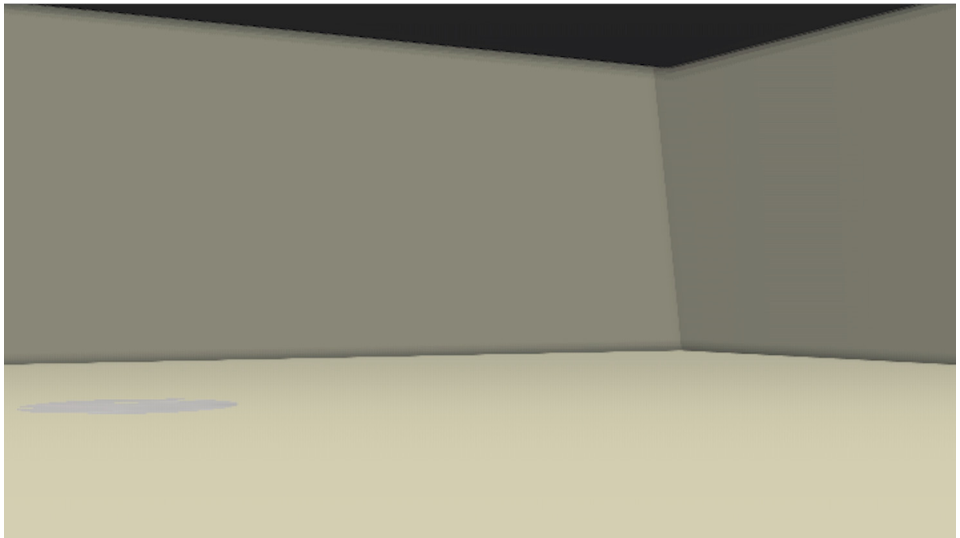
3D models are often **designed for viewing.**

Key Observations

3D models are often **designed for viewing.**

Key Observations

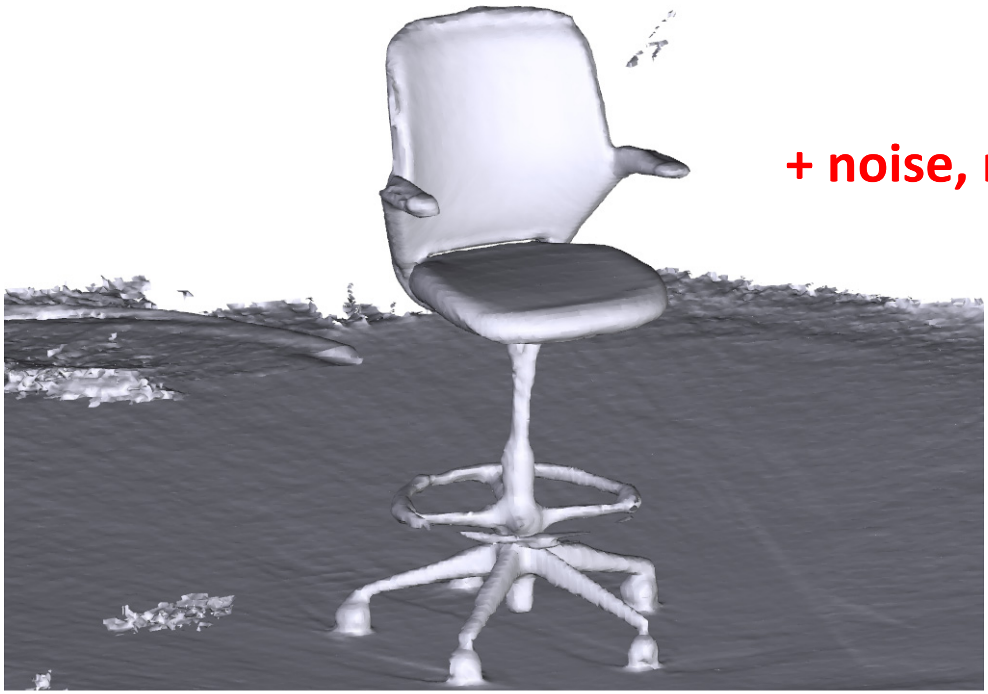
3D models are often **designed for viewing.**



Empty inside!

Key Observations

3D scans **capture the surface.**

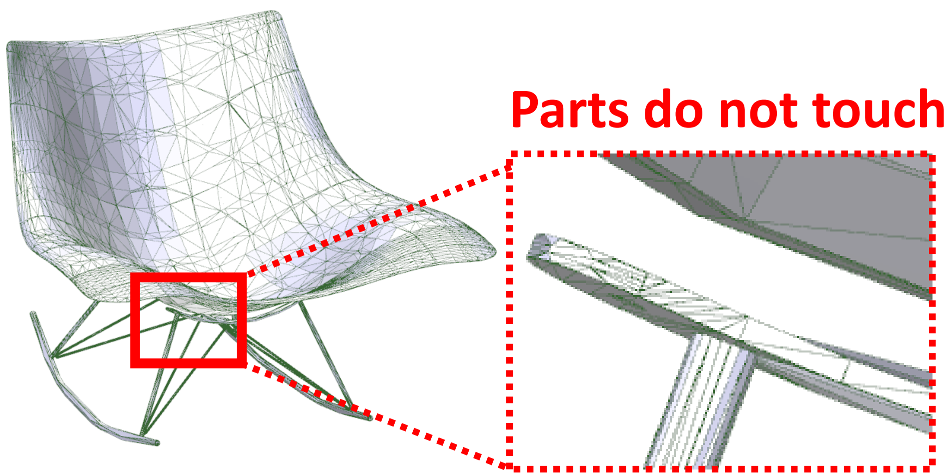


+ noise, missing regions etc

Key Observations

3D models are often **designed for viewing.**

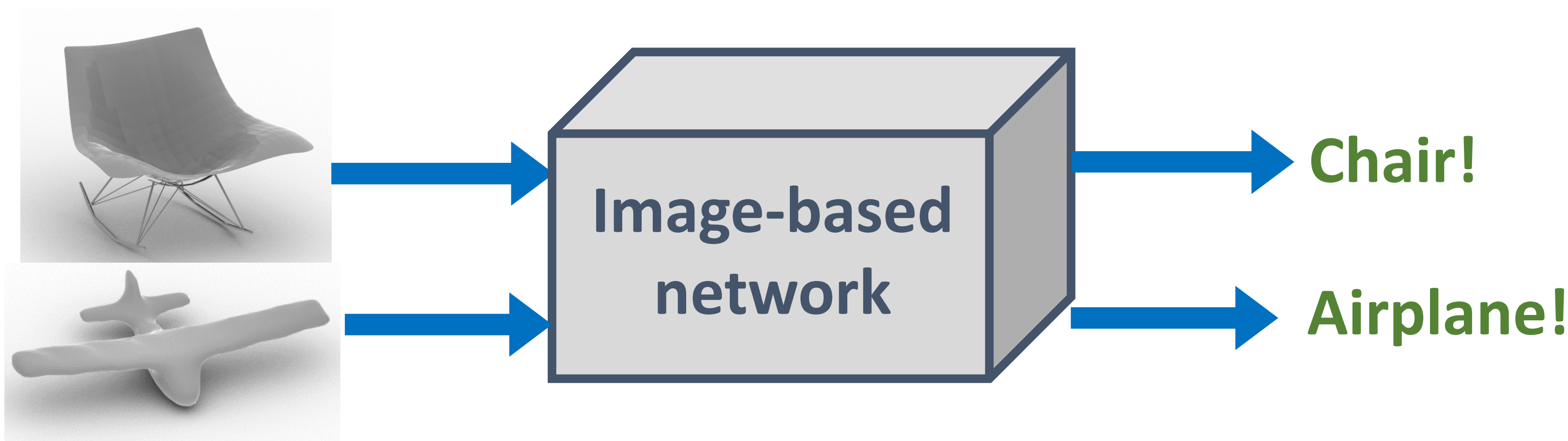
Parts do not touch!



(not easily noticeable to the viewer,
yet geometric implications on topology, connectedness...)

Key Observations

Shape renderings can be treated as **photos of objects** (without texture)



Shape renderings can be processed by powerful image-based architectures through transfer learning from massive image datasets.

(Su, Maji, Kalogerakis, Learned-Miller, ICCV 2015)

Key Idea

Deep architecture that combines **view-based convnets for part reasoning on rendered shape images** & **prob. graphical models for surface processing.**

Key Idea

Deep architecture that combines **view-based convnets for part reasoning on rendered shape images** & **prob. graphical models for surface processing.**

Key challenges:

- **Select views to avoid surface information loss & deal with occlusions**

Key Idea

Deep architecture that combines **view-based convnets for part reasoning on rendered shape images** & **prob. graphical models for surface processing.**

Key challenges:

- **Select views to avoid surface information loss & deal with occlusions**
- **Promote invariance under 3D shape rotations**

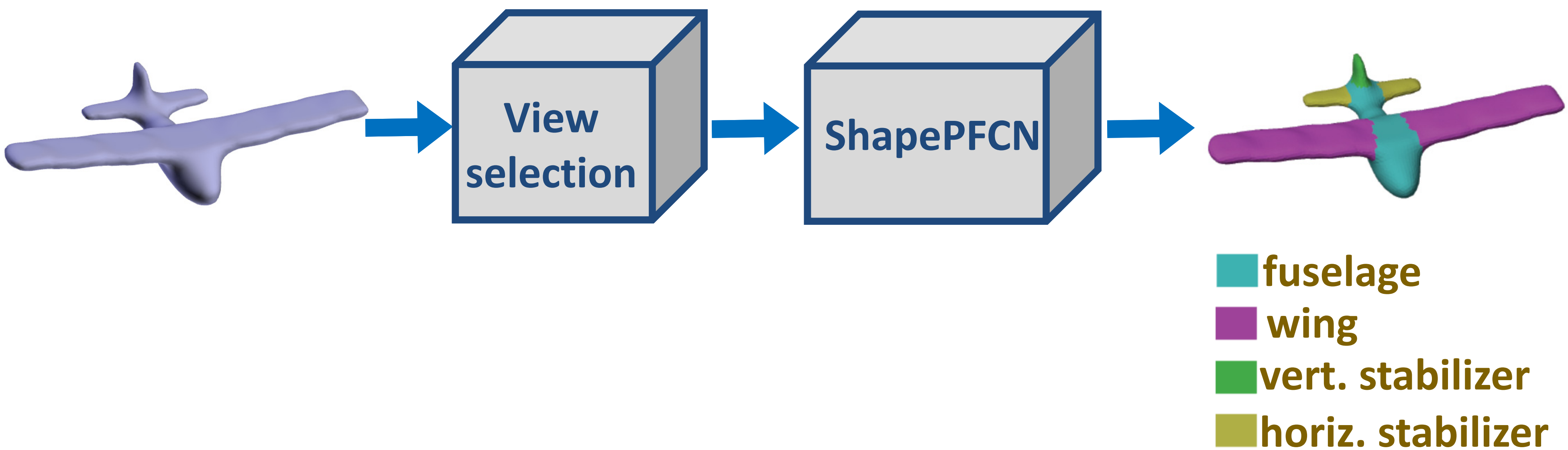
Key Idea

Deep architecture that combines **view-based convnets for part reasoning on rendered shape images** & **prob. graphical models for surface processing.**

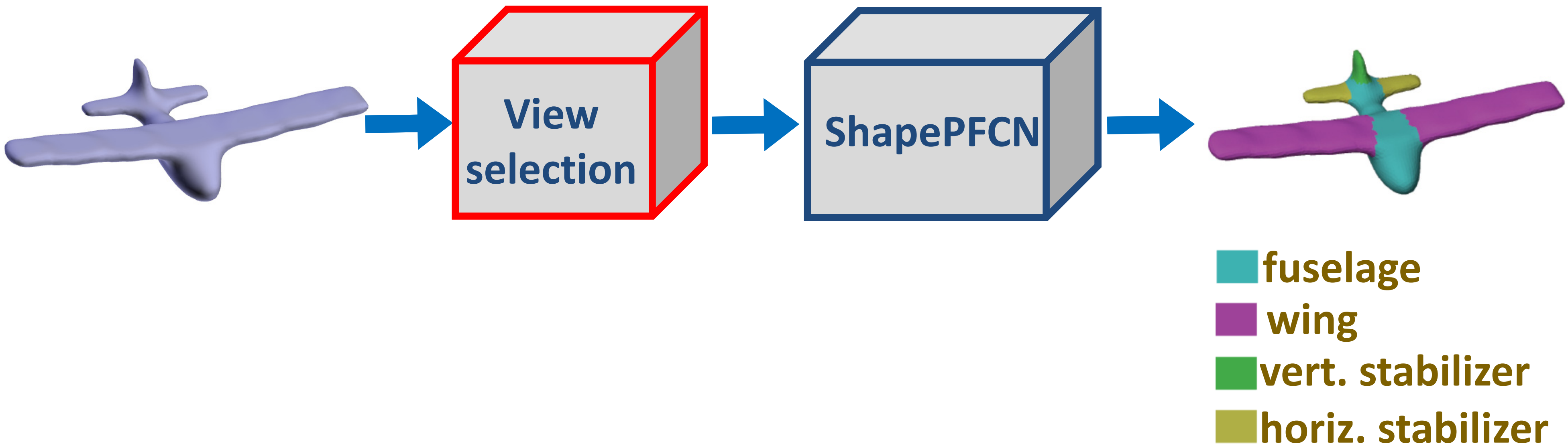
Key challenges:

- **Select views to avoid surface information loss & deal with occlusions**
- **Promote invariance under 3D shape rotations**
- **Joint reasoning about parts across multiple views + surface**

Pipeline


View selection
ShapePFCN
fuselage
wing
vert. stabilizer
horiz. stabilizer

Pipeline

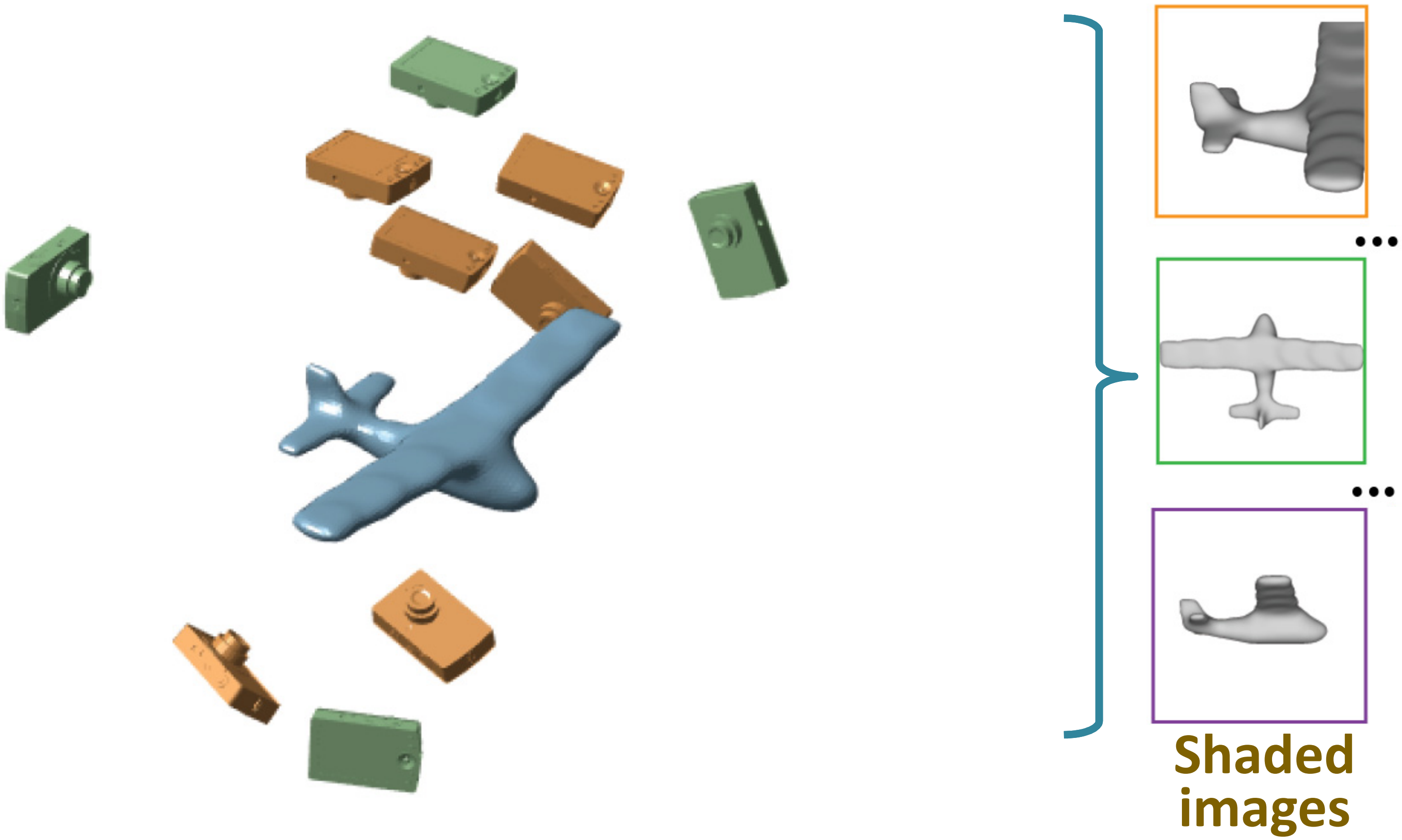

View selection
ShapePFCN
fuselage
wing
vert. stabilizer
horiz. stabilizer

Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.

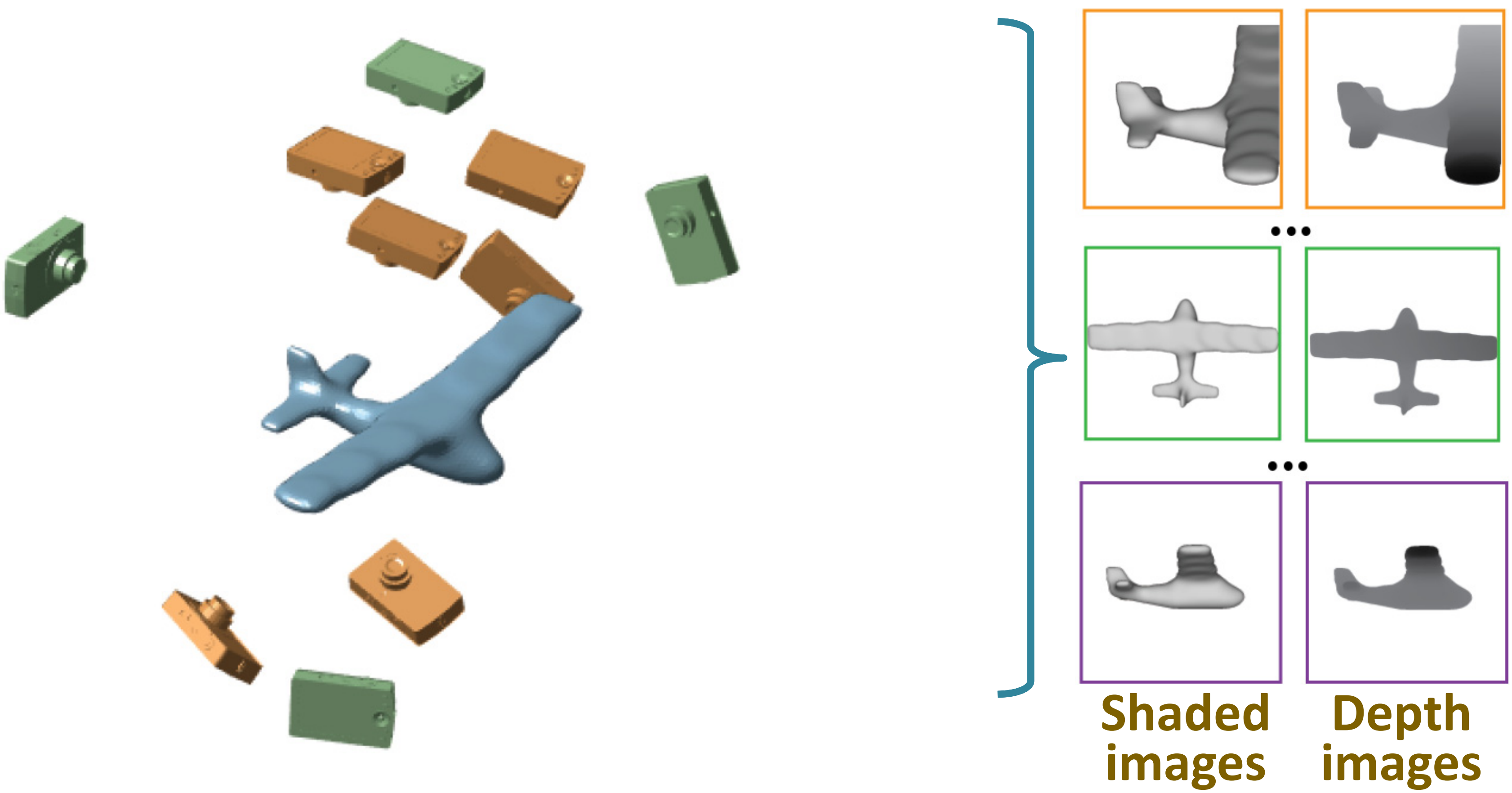
Input: shape as a collection of rendered views

Render **shaded images** (normal dot view vector) encoding surface normals.



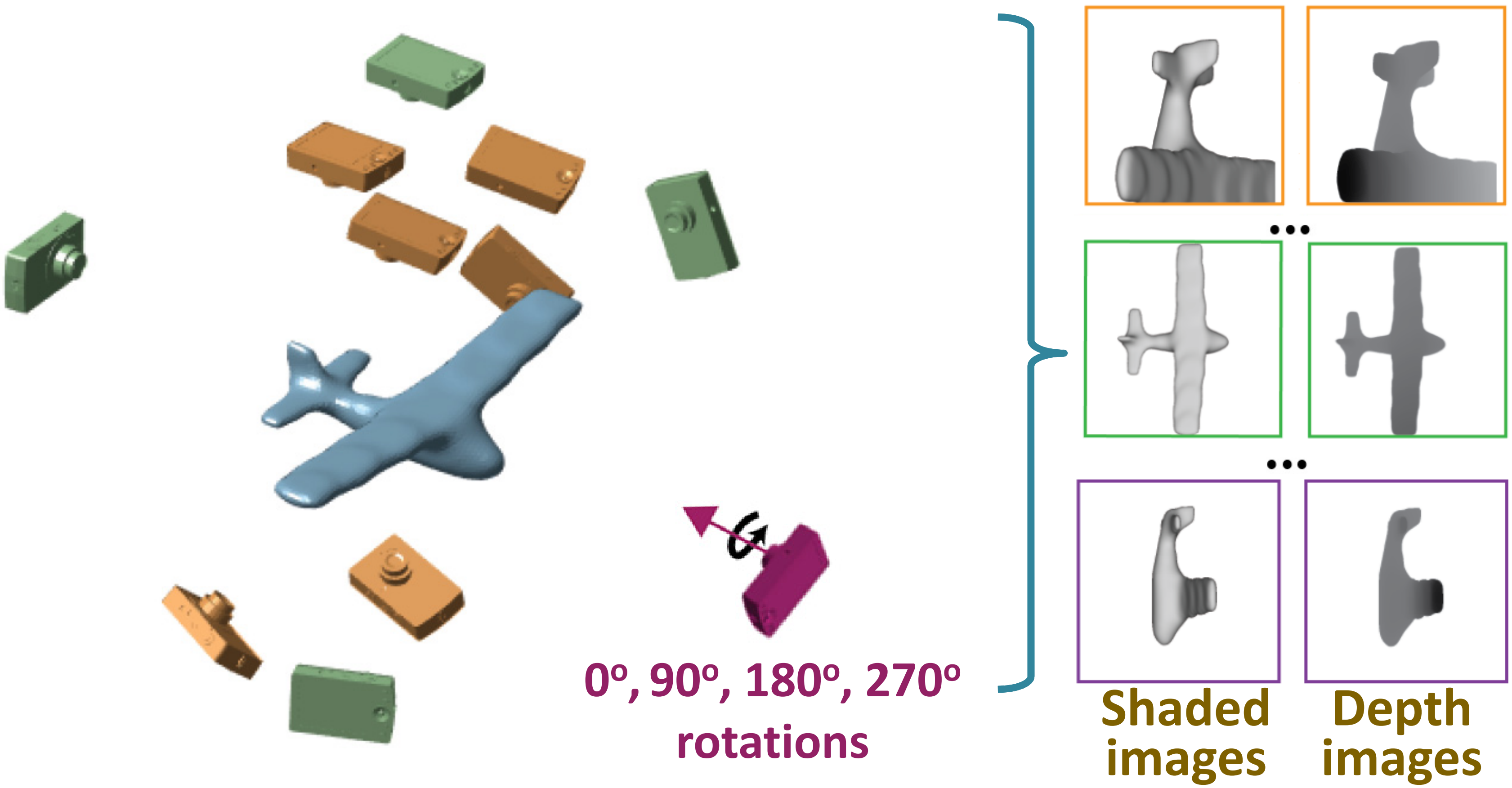
Input: shape as a collection of rendered views

Render also **depth images** encoding surface position relative to the camera.



Input: shape as a collection of rendered views

Perform in-plane camera rotations for **rotational invariance.**

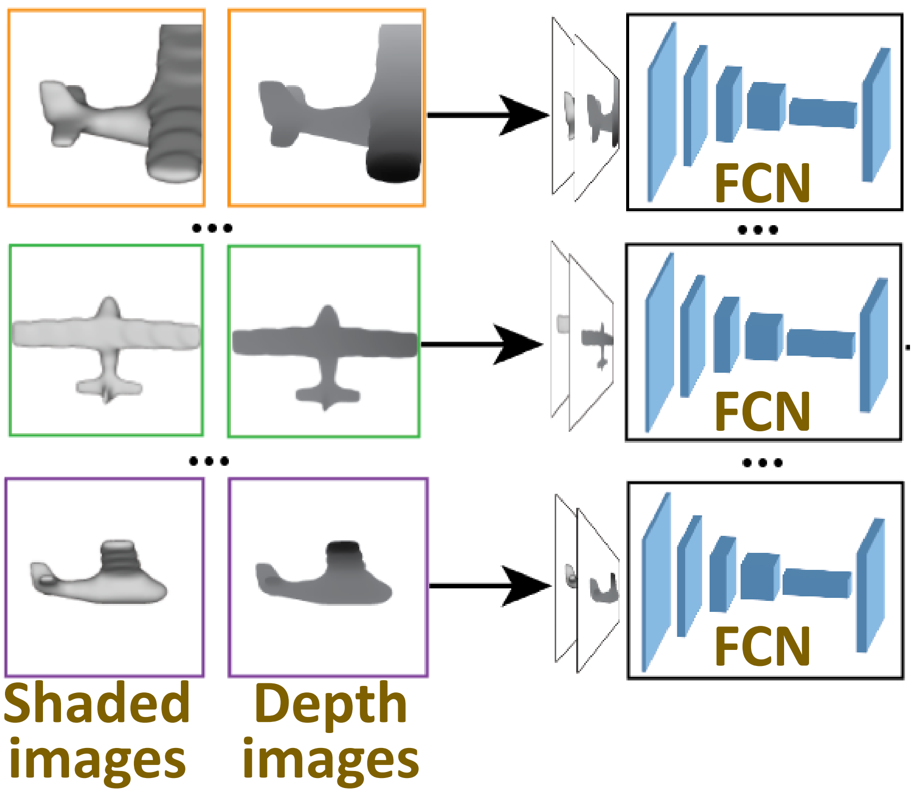


Projective convnet architecture

Each **pair of depth & shaded images** is processed by a FCN.

Views are **not ordered** (**no view correspondence** across shapes).

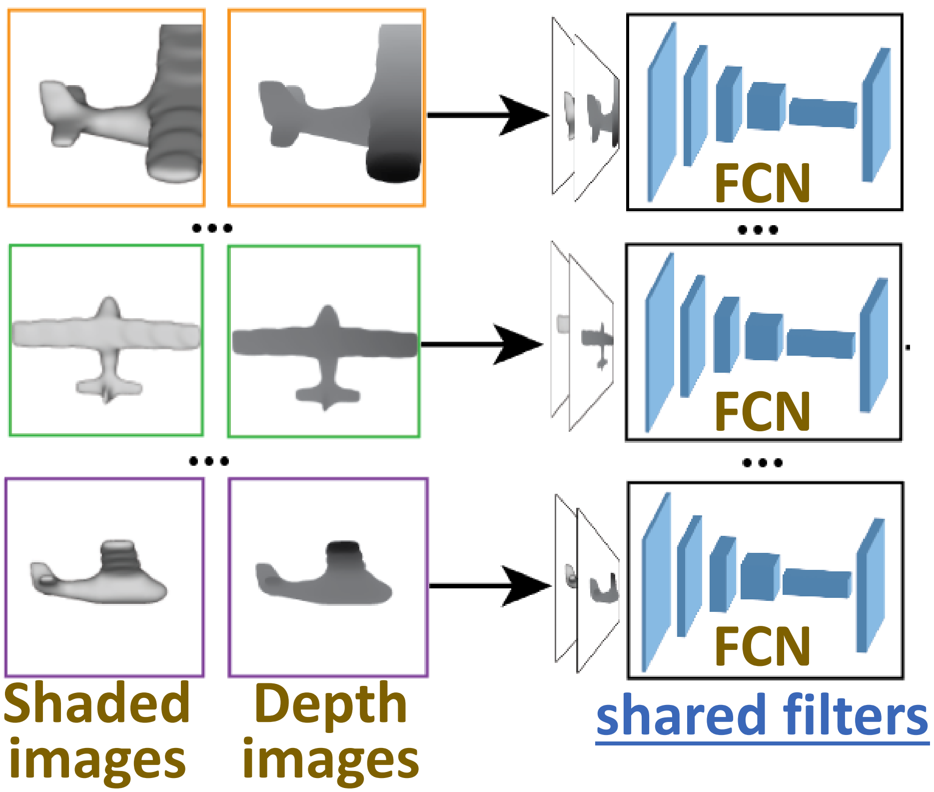
[Long, Shelhamer, and Darrell 2015]



Projective convnet architecture

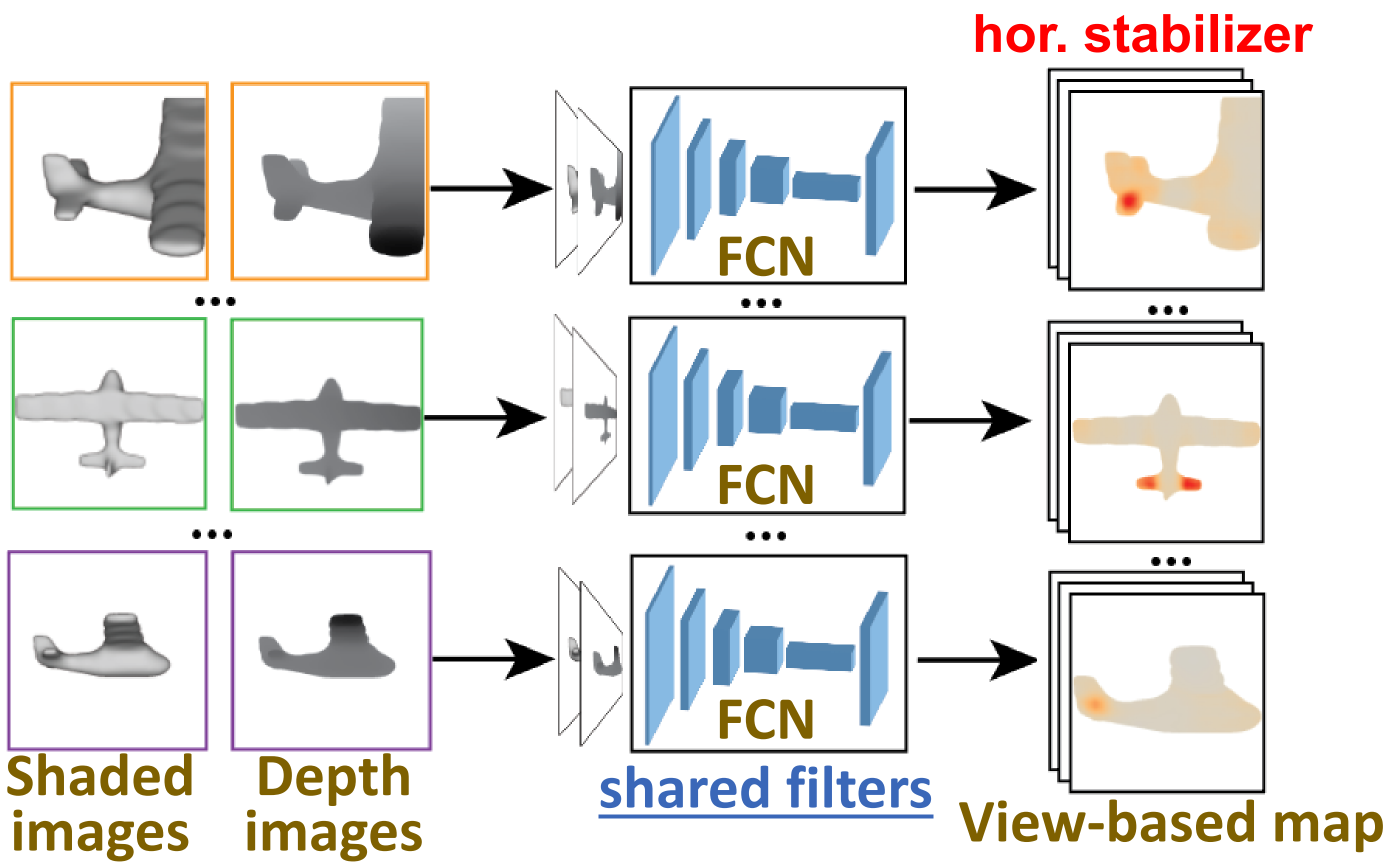
Each **pair of depth & shaded images** is processed by a FCN. [Long, Shelhamer, and Darrell 2015]

Views are **not ordered** (**no view correspondence** across shapes).



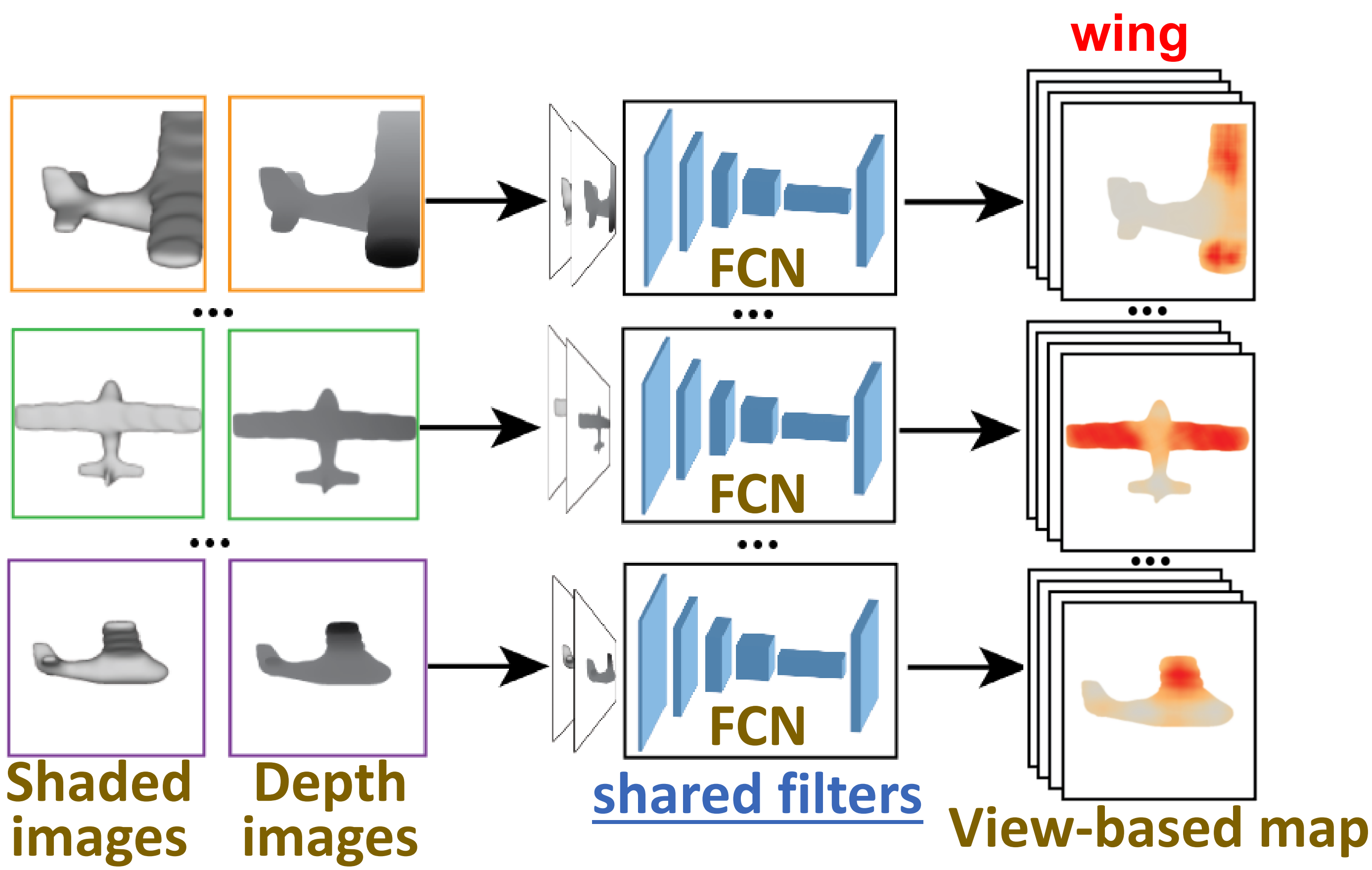
Projective convnet architecture

The output of each FCN branch is a view-based **confidence map per part label.**



Projective convnet architecture

The output of each FCN branch is a view-based **confidence map per part label.**



Projective convnet architecture

Aggregate & **project** the image confidence maps from all views **on the surface.**

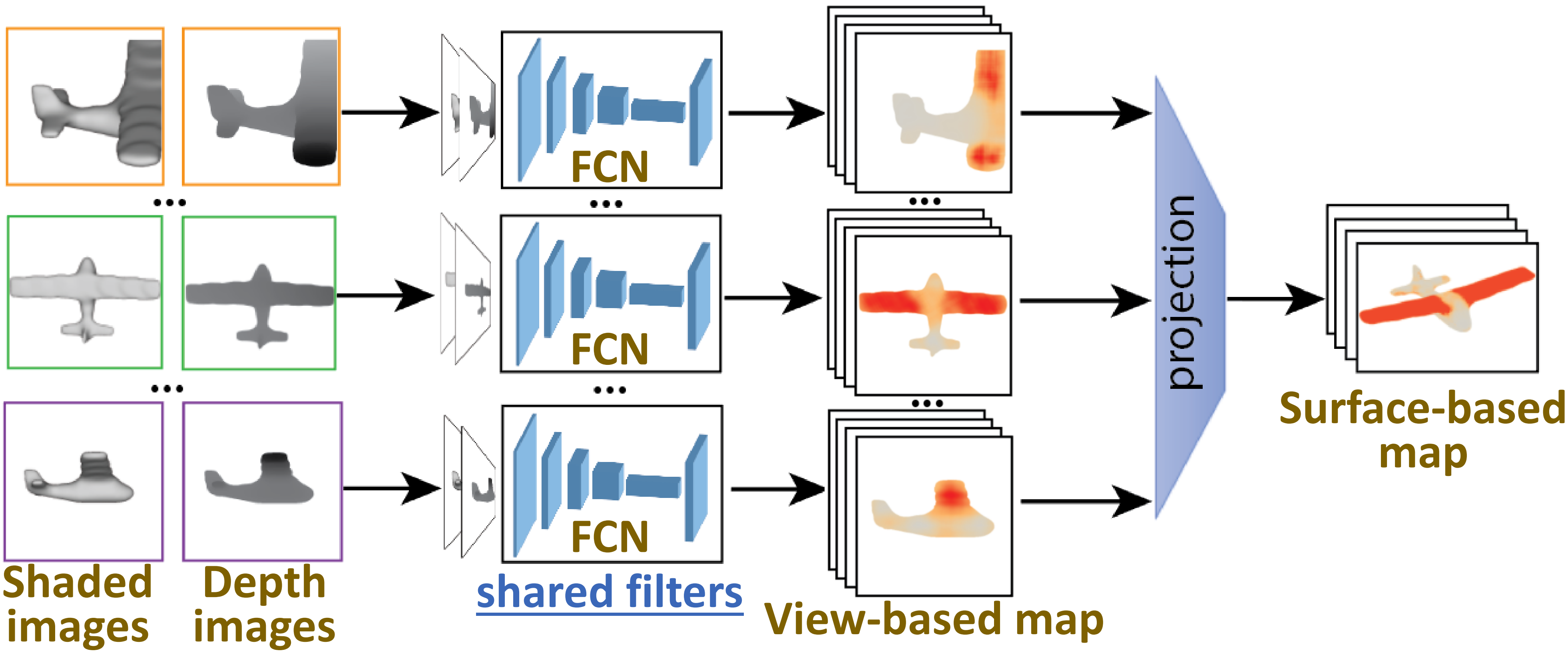


Image2Surface projection layer

For each surface element (triangle), find all pixels that include it in all views.

Surface confidence: use **max of these pixel confidences** per label.

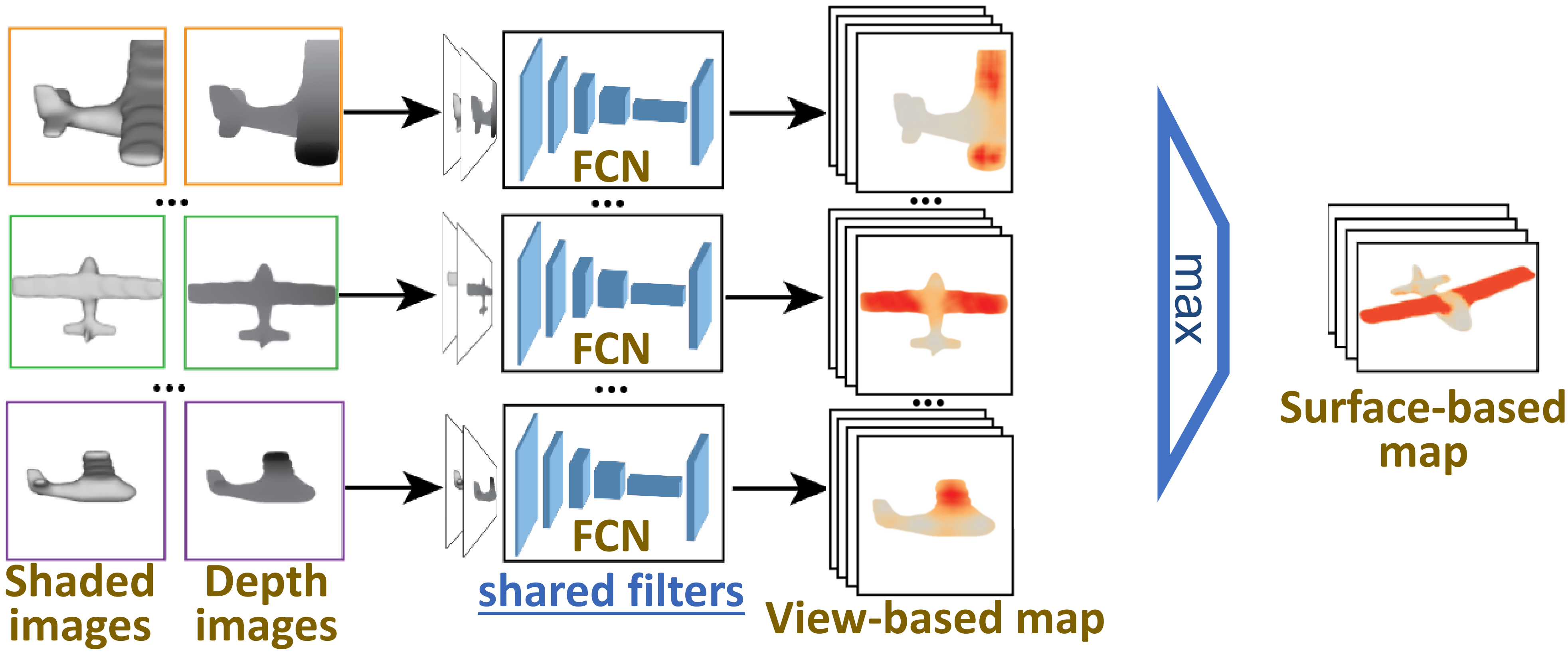


Image2Surface projection layer

For each surface element (triangle), find all pixels that include it in all views.

Surface confidence: use **max of these pixel confidences** per label.

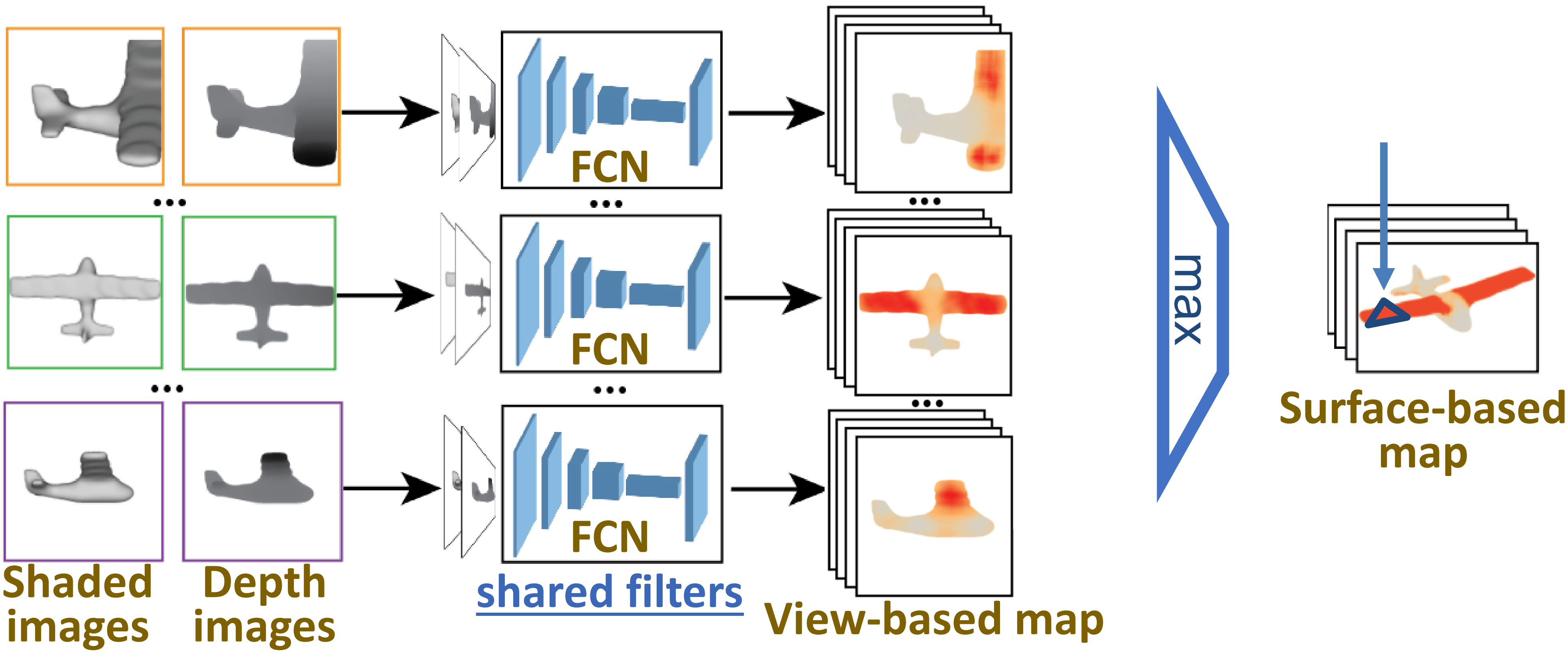
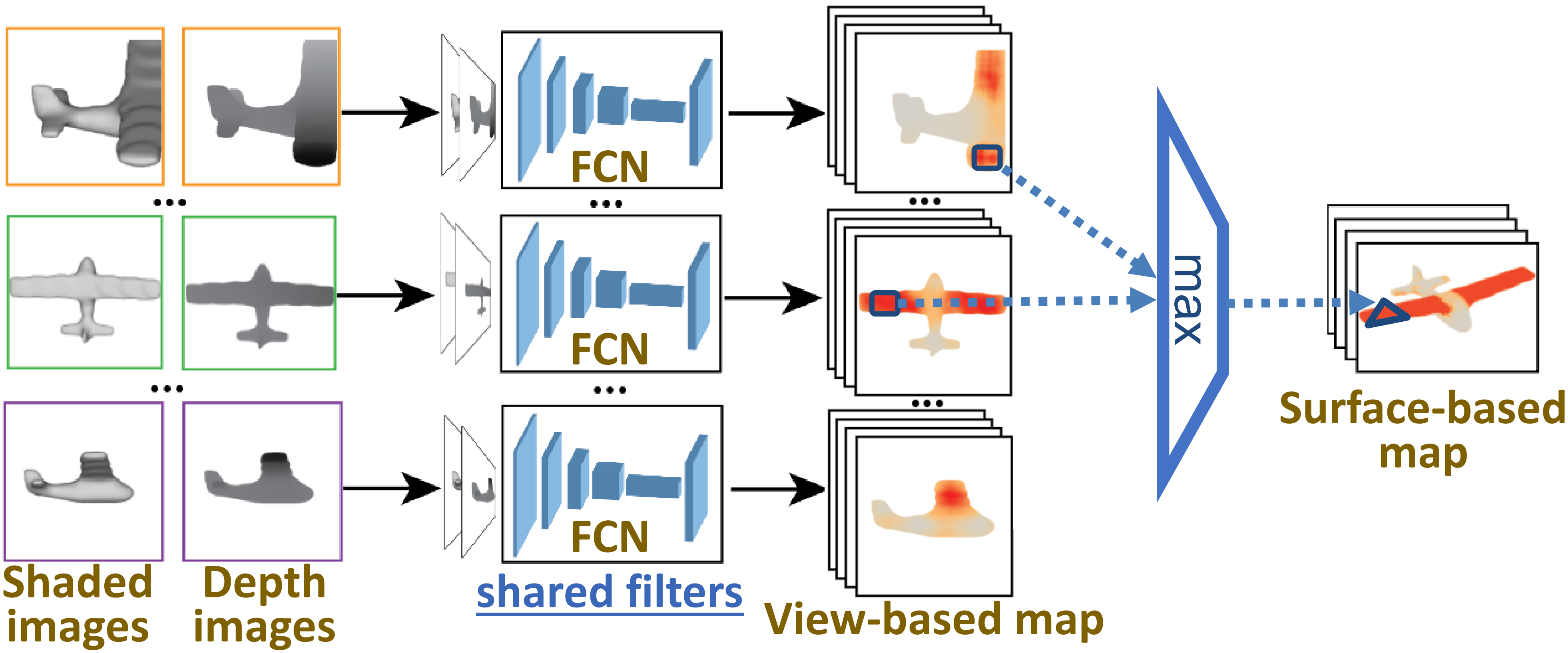


Image2Surface projection layer

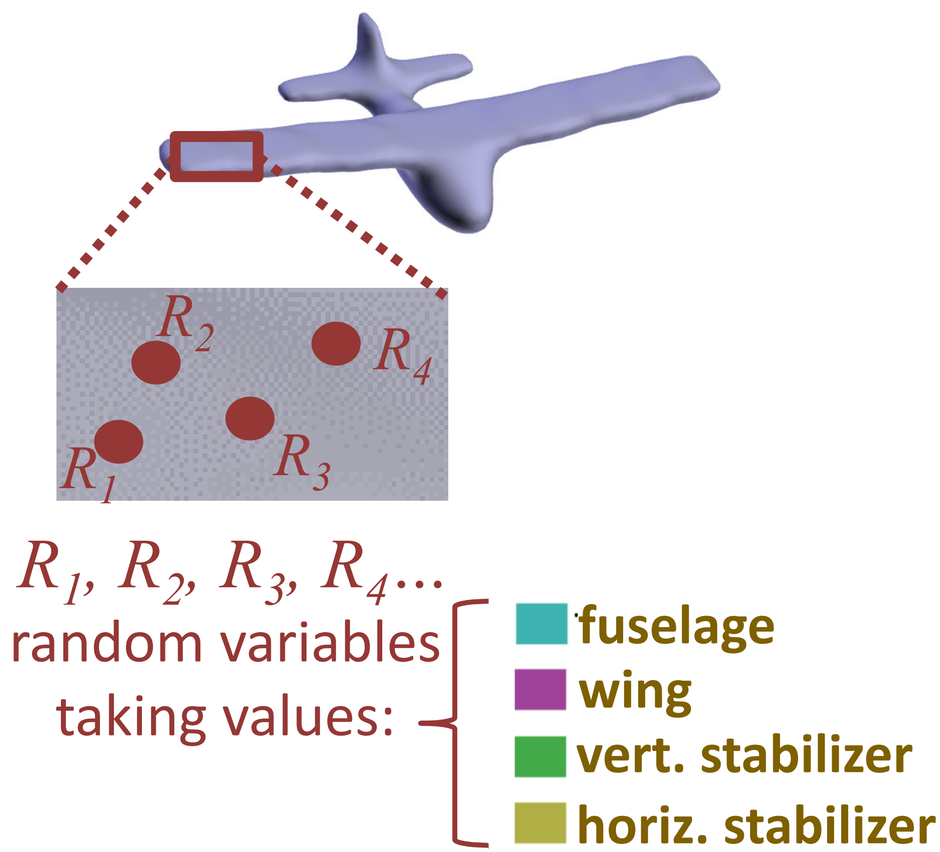
For each surface element (triangle), find all pixels that include it in all views.

Surface confidence: use **max of these pixel confidences** per label.



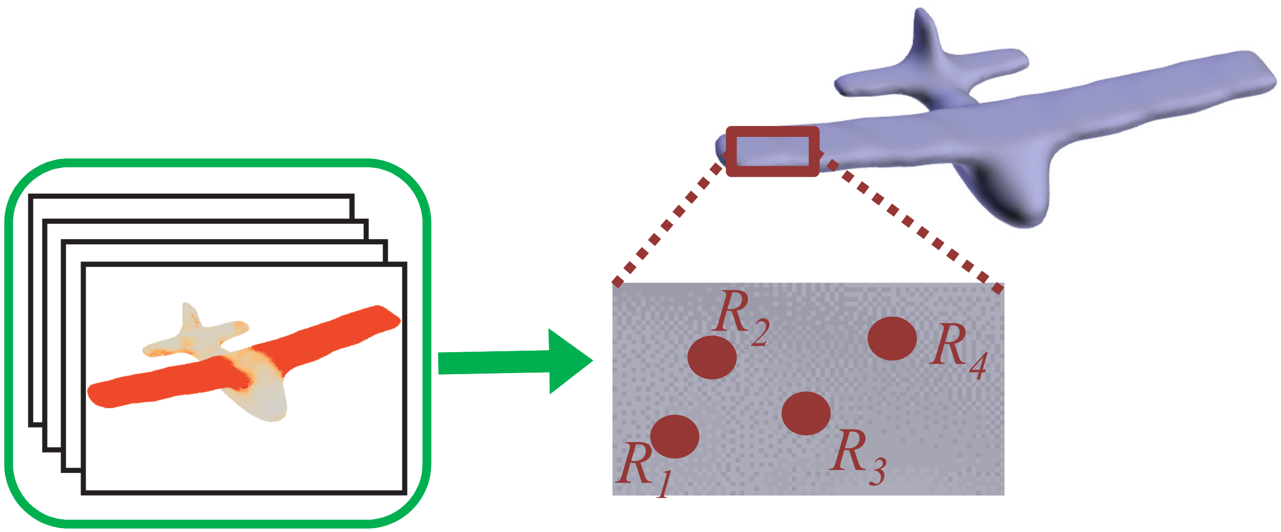
CRF layer for spatially coherent labeling

Last layer performs **inference in a probabilistic model defined on the surface.**



CRF layer for spatially coherent labeling

Conditional Random Field: unary factors based on surface-based confidences



$$P(R_1, R_2, R_3, R_4 \ldots \mid \textbf{shape}) = \frac{1}{Z} \boxed{\prod_{f=1..n} P(R_f \mid \textbf{views})} \prod_{f,f'} P(R_f, R_{f'} \mid \textbf{surface})$$

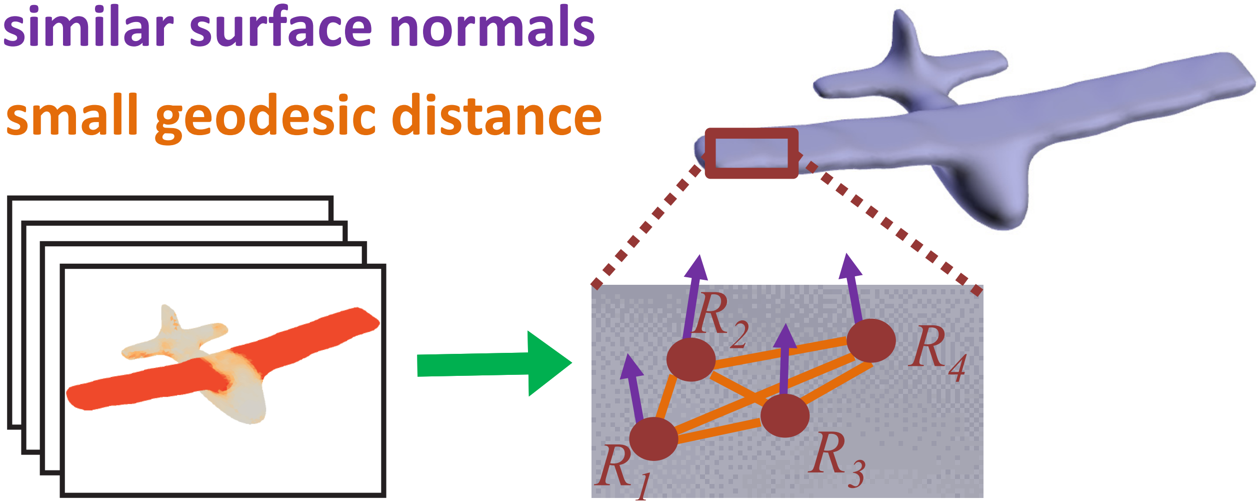
Unary factors
(FCN confidences)

Projective convnet architecture: CRF layer

Pairwise terms **favor same label** for triangles with:

(a) **similar surface normals**

(b) **small geodesic distance**

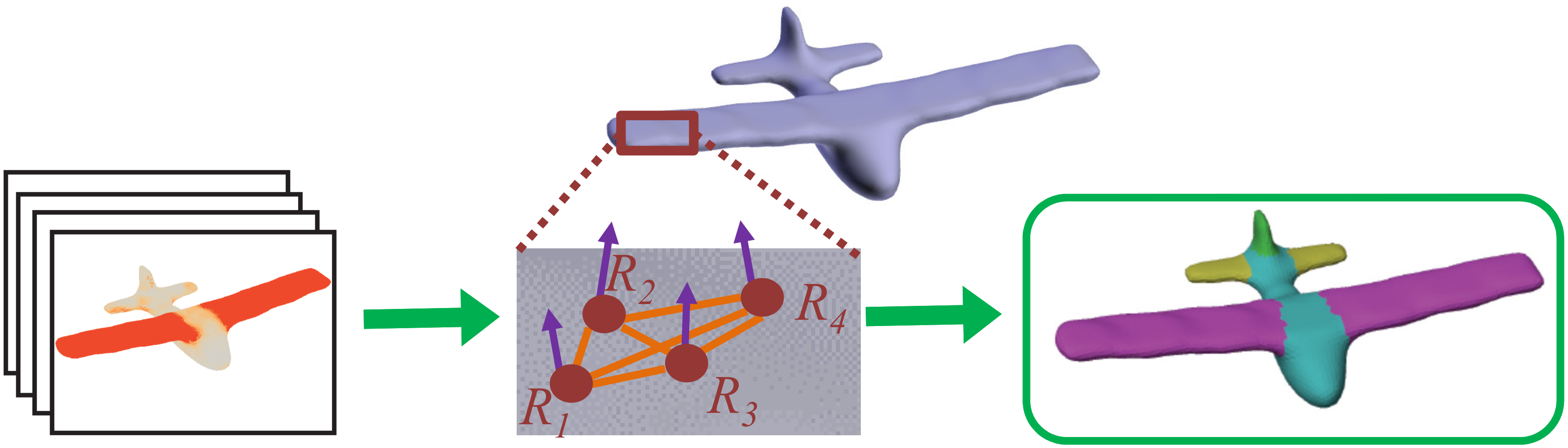


$$P(R_1, R_2, R_3, R_4 \ldots \mid \textbf{shape}) = \frac{1}{Z} \prod_{f=1..n} P(R_f \mid \textbf{views}) \boxed{\prod_{f,f'} P(R_f, R_{f'} \mid \textbf{surface})}$$

Pairwise factors (geodesic+normal distance)

Projective convnet architecture: CRF layer

Infer **most likely joint assignment** to all surface random variables (mean-field)

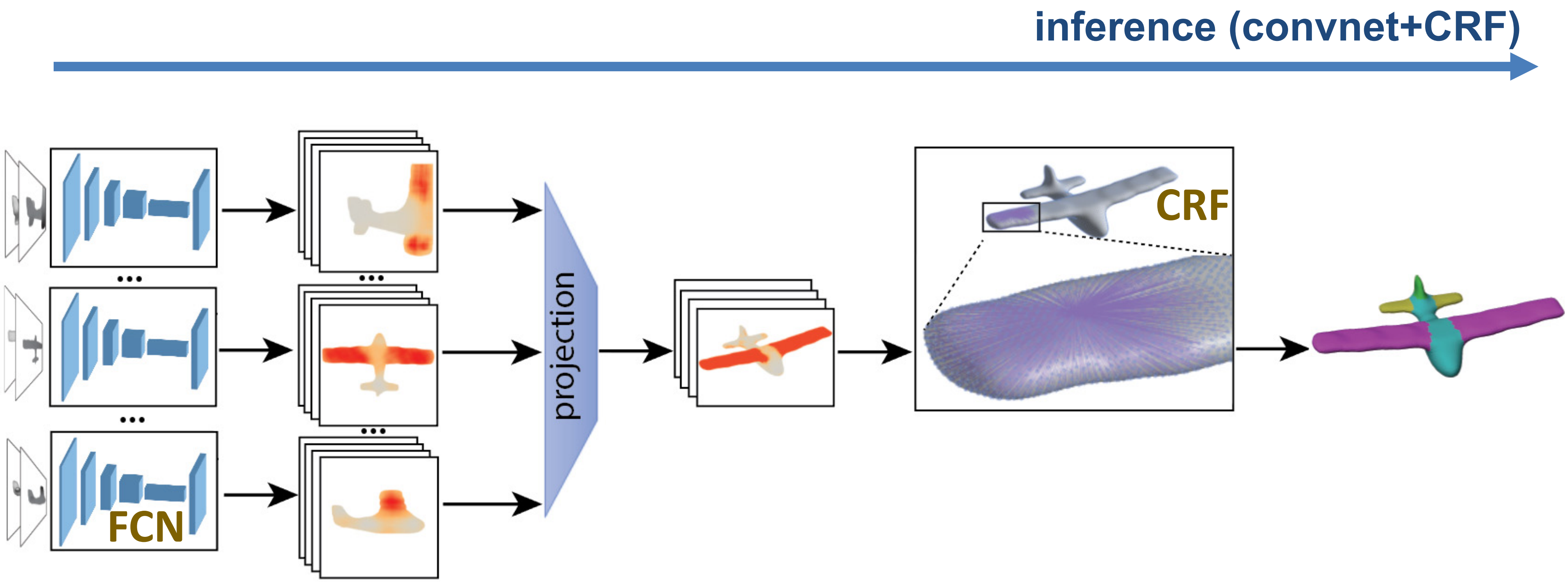


max

$$\boxed{P(R_1, R_2, R_3, R_4 \ldots \mid \textbf{shape})} = \frac{1}{Z} \prod_{f=1..n} P(R_f \mid \textbf{views}) \prod_{f,f'} P(R_f, R_{f'} \mid \textbf{surface})$$

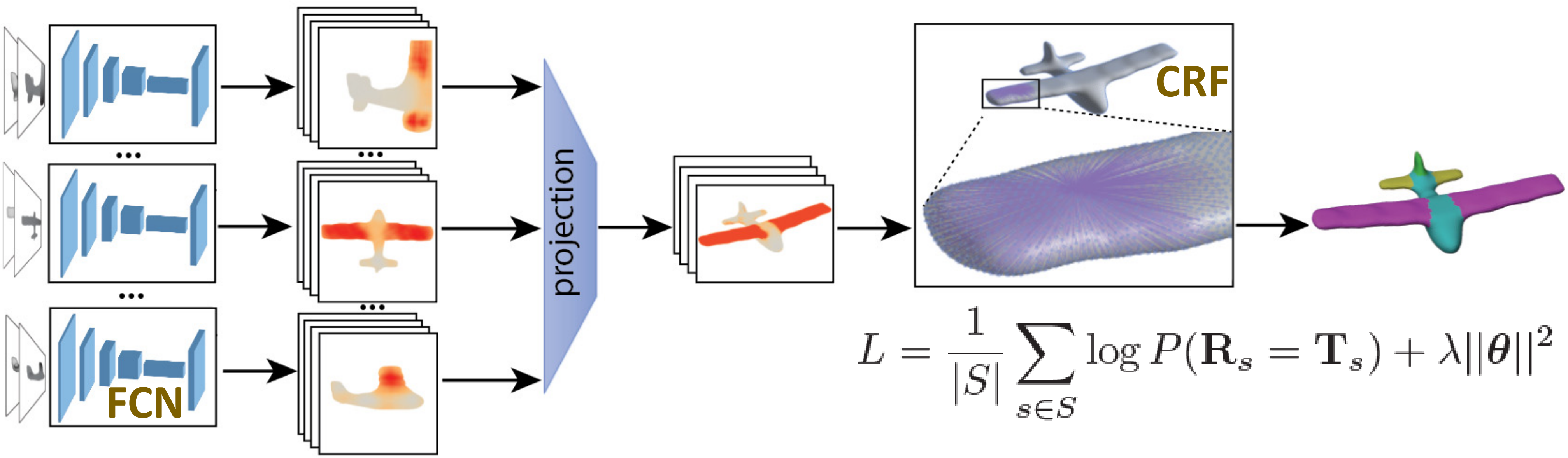
MAP assignment (mean–field inference)

Forward pass



Training

The architecture is trained **end-to-end** with analytic gradients.



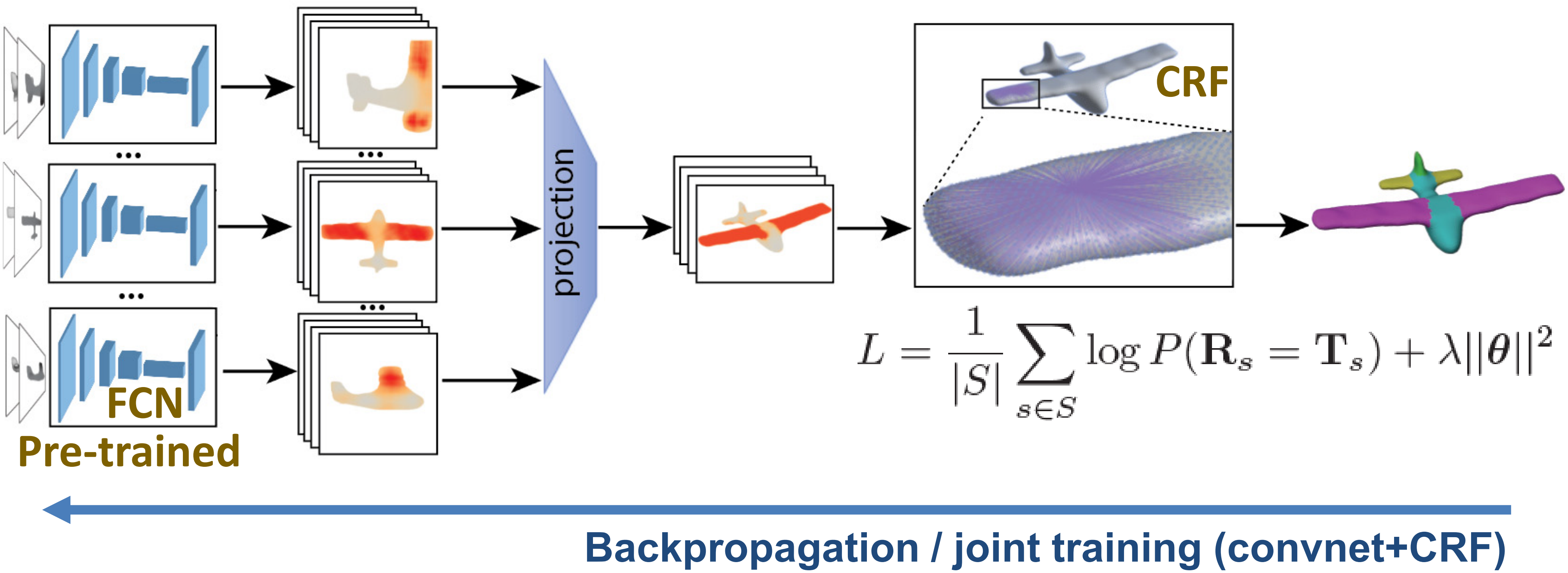
$$L = \frac{1}{|S|} \sum_{s \in S} \log P(\mathbf{R}_s = \mathbf{T}_s) + \lambda ||\boldsymbol{\theta}||^2$$

Backpropagation / joint training (convnet+CRF)

Training

The architecture is trained **end-to-end** with analytic gradients.

Training starts from a **pretrained image-based net** (VGG16), then **fine-tune.**



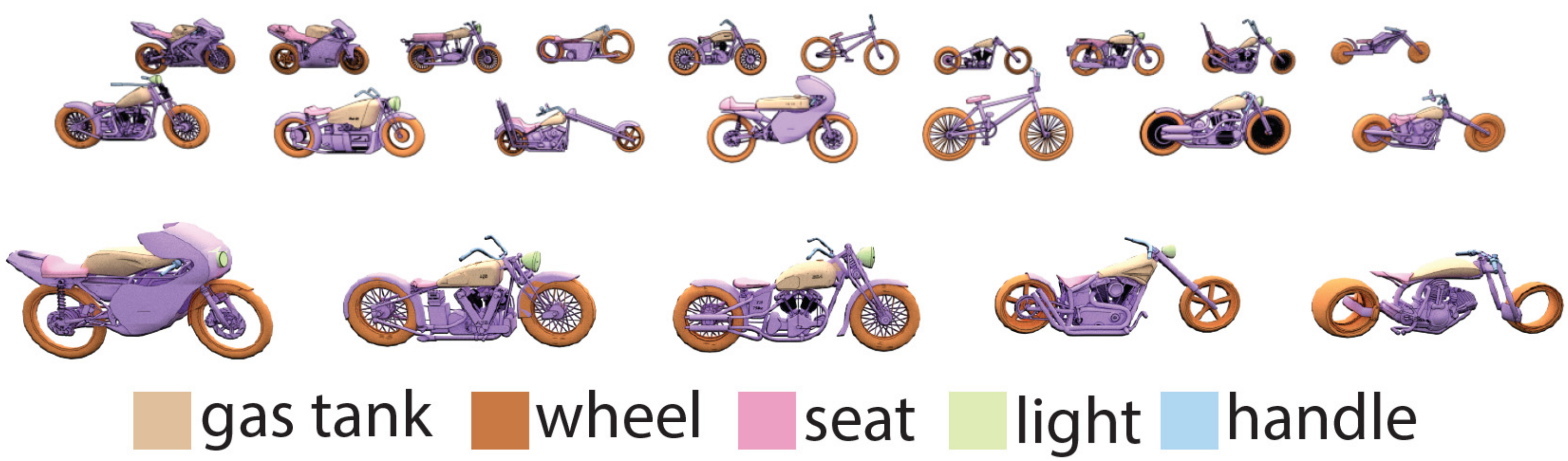
$$L = \frac{1}{|S|} \sum_{s \in S} \log P(\mathbf{R}_s = \mathbf{T}_s) + \lambda ||\theta||^2$$

Dataset used in experiments

Evaluation on **ShapeNet + LPSB + COSEG** (46 classes of shapes)

50% used for training / **50%** used for test split **per Shapenet category**

Max 250 shapes for training. No assumption on shape orientation.



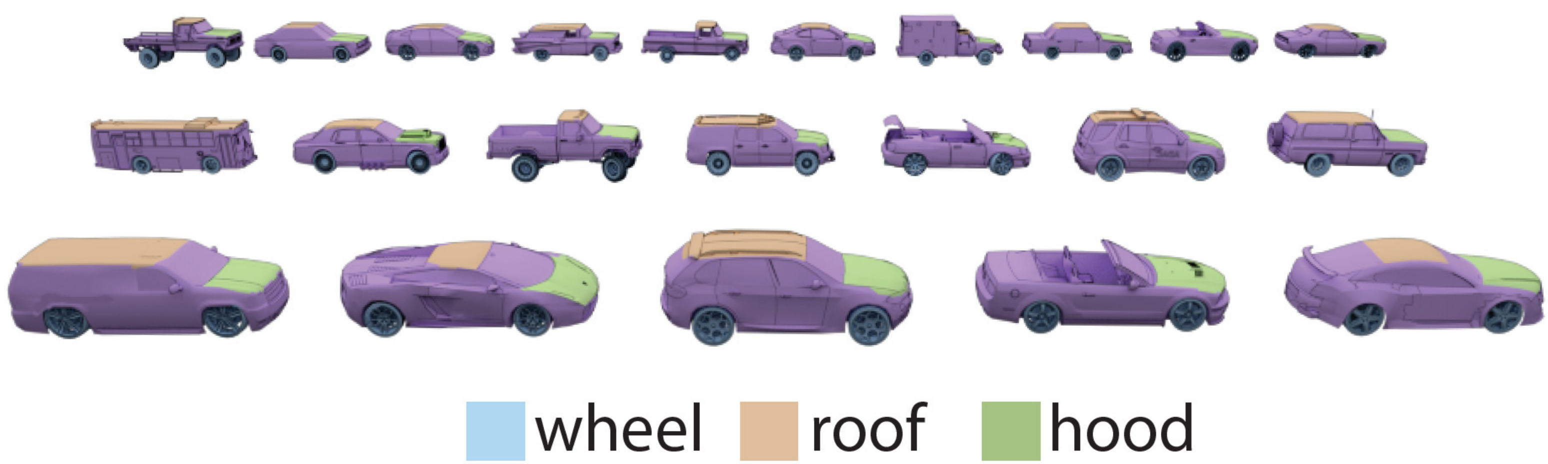
[Yi et al. 2016]

Dataset used in experiments

Evaluation on **ShapeNet + LPSB + COSEG** (46 classes of shapes)

50% used for training / **50%** used for test split **per Shapenet category**

Max 250 shapes for training. No assumption on shape orientation.



[Yi et al. 2016]

Results

Labeling accuracy on ShapeNet test dataset:

ShapeBoost	Guo et al.	ShapePFCN
81.2	80.6	**87.5**

Results

Labeling accuracy on ShapeNet test dataset:

	ShapeBoost	Guo et al.	ShapePFCN
	81.2	80.6	**87.5**
Ignore easy classes (2 or 3 part labels)	76.8	76.8	**84.7**

8% improvement in labeling accuracy for complex categories (vehicles, furniture)

Results

Labeling accuracy on ShapeNet test dataset:

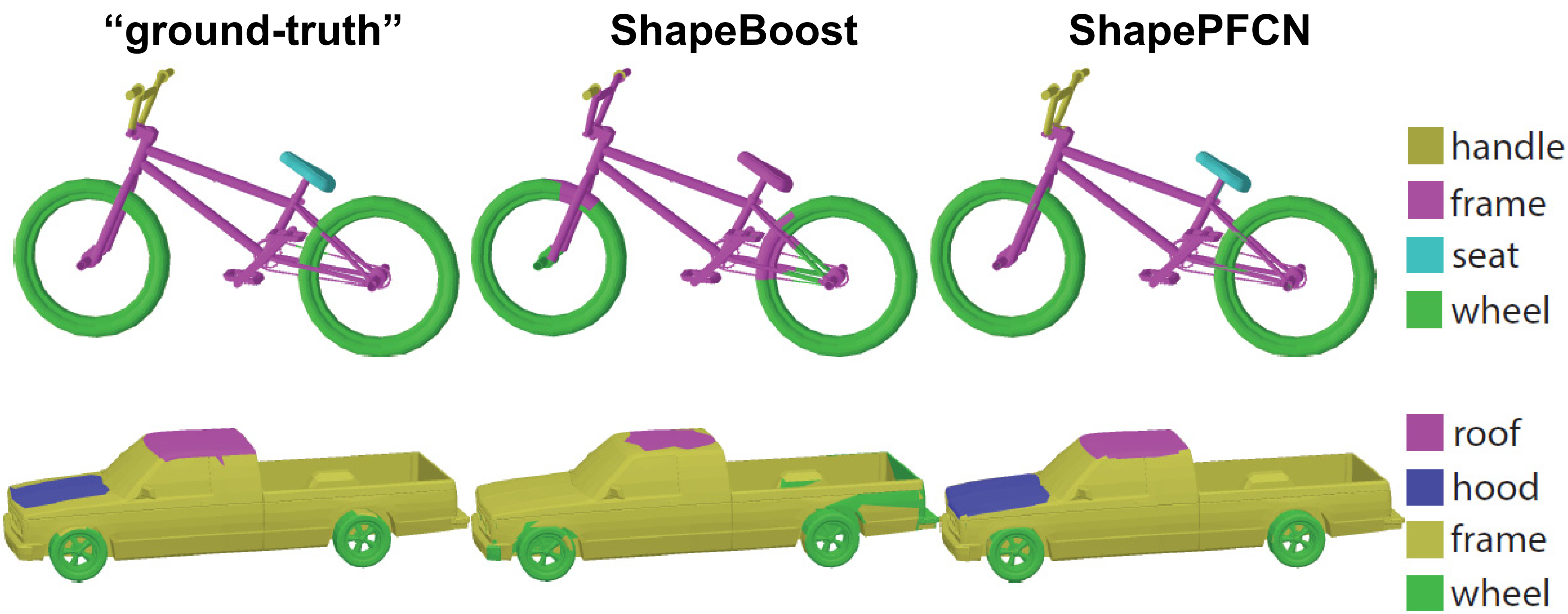
ShapeBoost	Guo et al.	ShapePFCN
81.2	80.6	**87.5**
76.8	76.8	**84.7**

Ignore easy classes (2 or 3 part labels)

8% improvement in labeling accuracy for complex categories (vehicles, furniture)

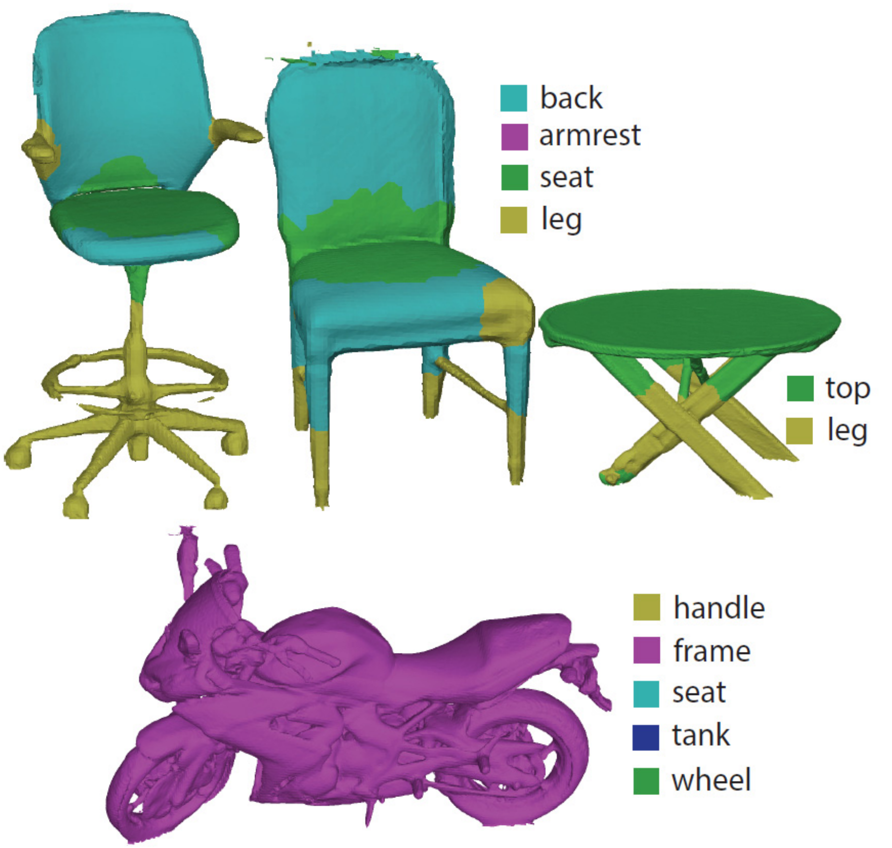
Labeling accuracy on LPSB+COSEG test dataset:

ShapeBoost	Guo et al.	ShapePFCN
84.2	82.1	**92.2**

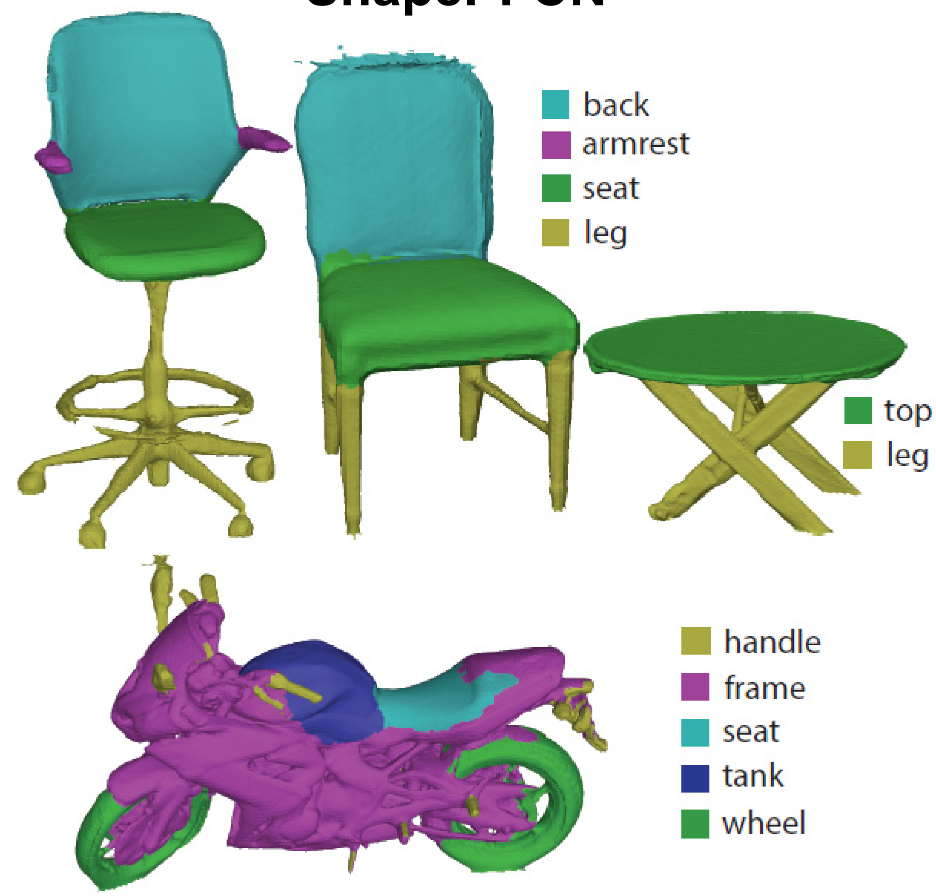


“ground-truth”
ShapeBoost
ShapePFCN
handle
frame
seat
wheel
roof
hood
frame
wheel

ShapeBoost



ShapePFCN



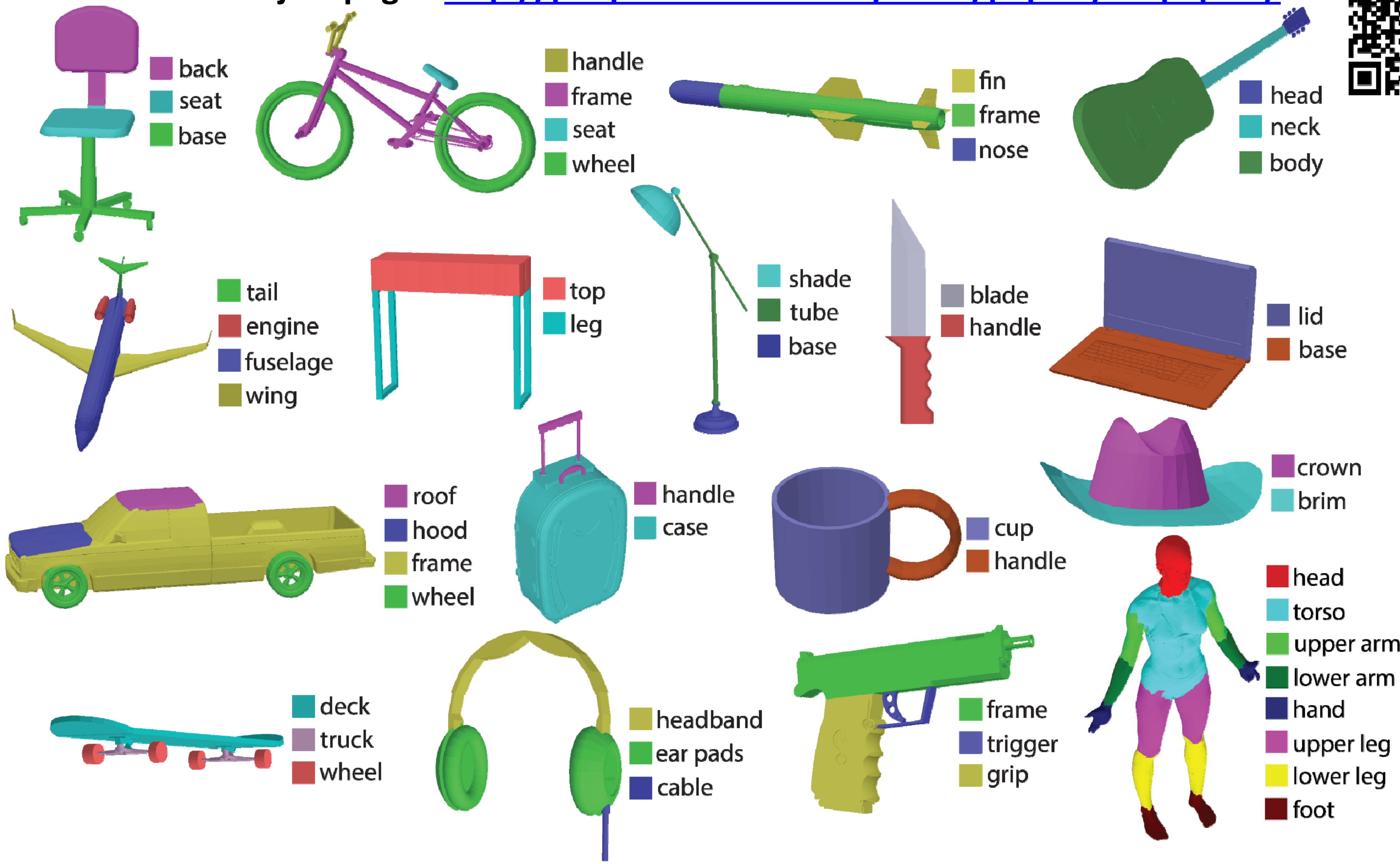
Object scans from “A Large Dataset of Object Scans” Choi et al. 2016

Summary

- Deep architecture combining **view-based FCN** & **surface-based CRF**
- **Multi-scale view selection** to avoid loss of surface information
- Transfer learning from **massive image datasets**
- **Robust** to geometric representation artifacts

Acknowledgements: NSF (CHS-1422441, CHS-1617333, IIS- 1617917)
Experiments were performed in the **UMass GPU cluster (400 GPUs!)** obtained under a grant by the MassTech Collaborative

Project page: http://people.cs.umass.edu/~kalo/papers/shapepfcn/



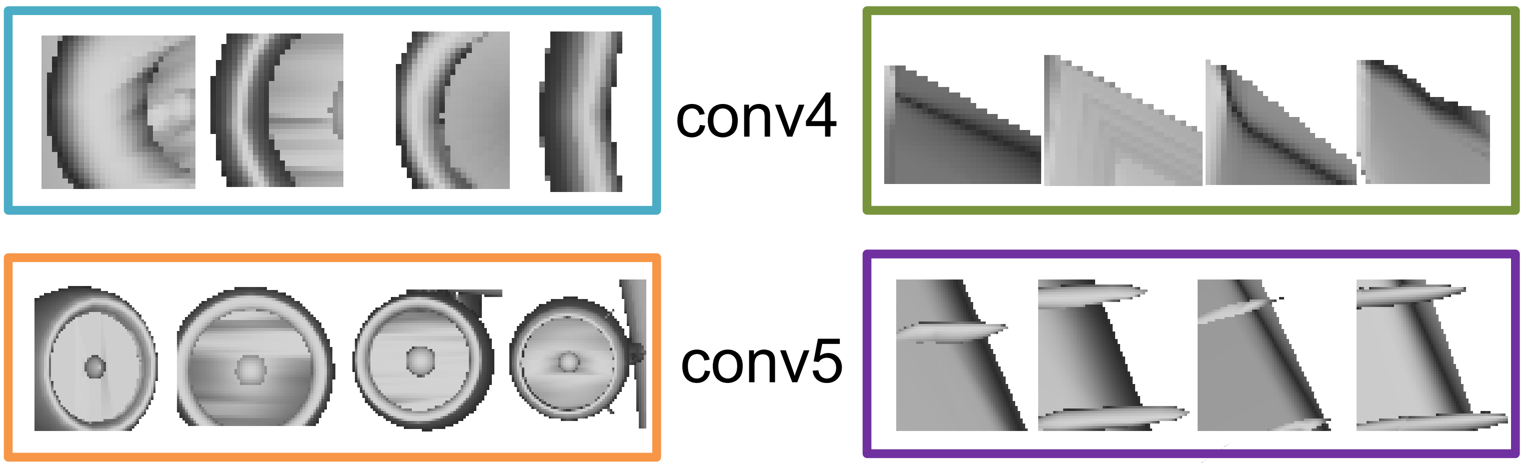
Auxiliary slides

What are the filters doing?

Activated in the presence of certain patterns of surface patches

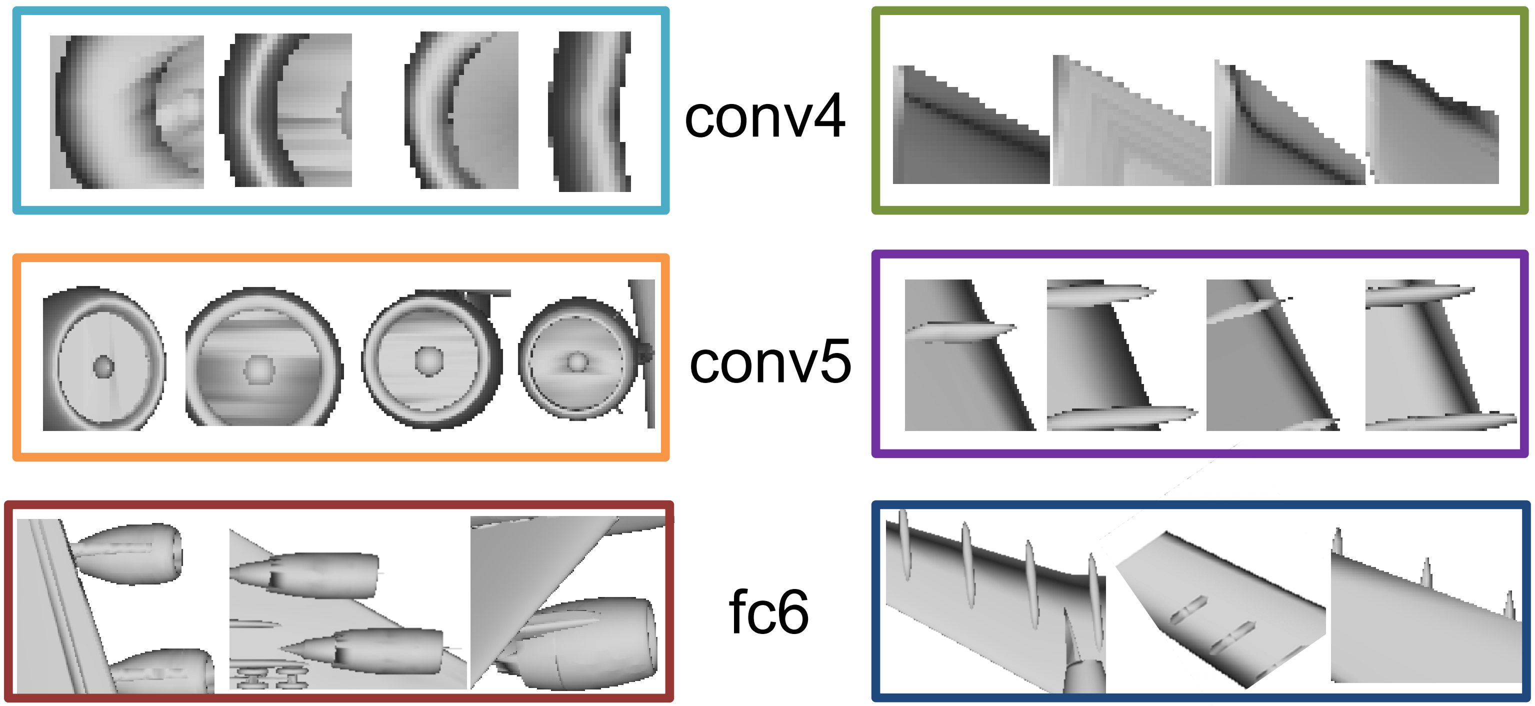
What are the filters doing?

Activated in the presence of certain patterns of surface patches



What are the filters doing?

Activated in the presence of certain patterns of surface patches

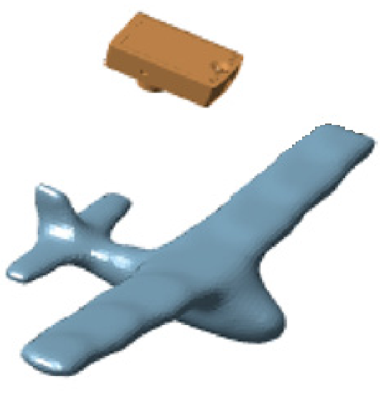


Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.

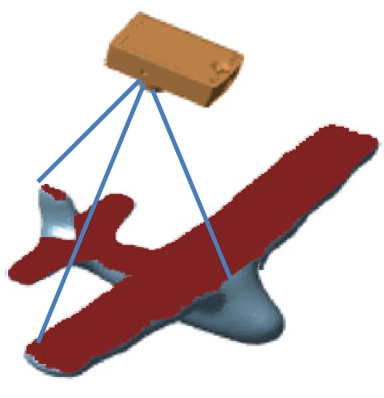
Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.



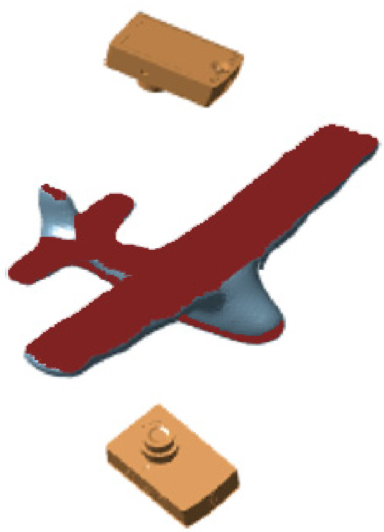
Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.



Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.

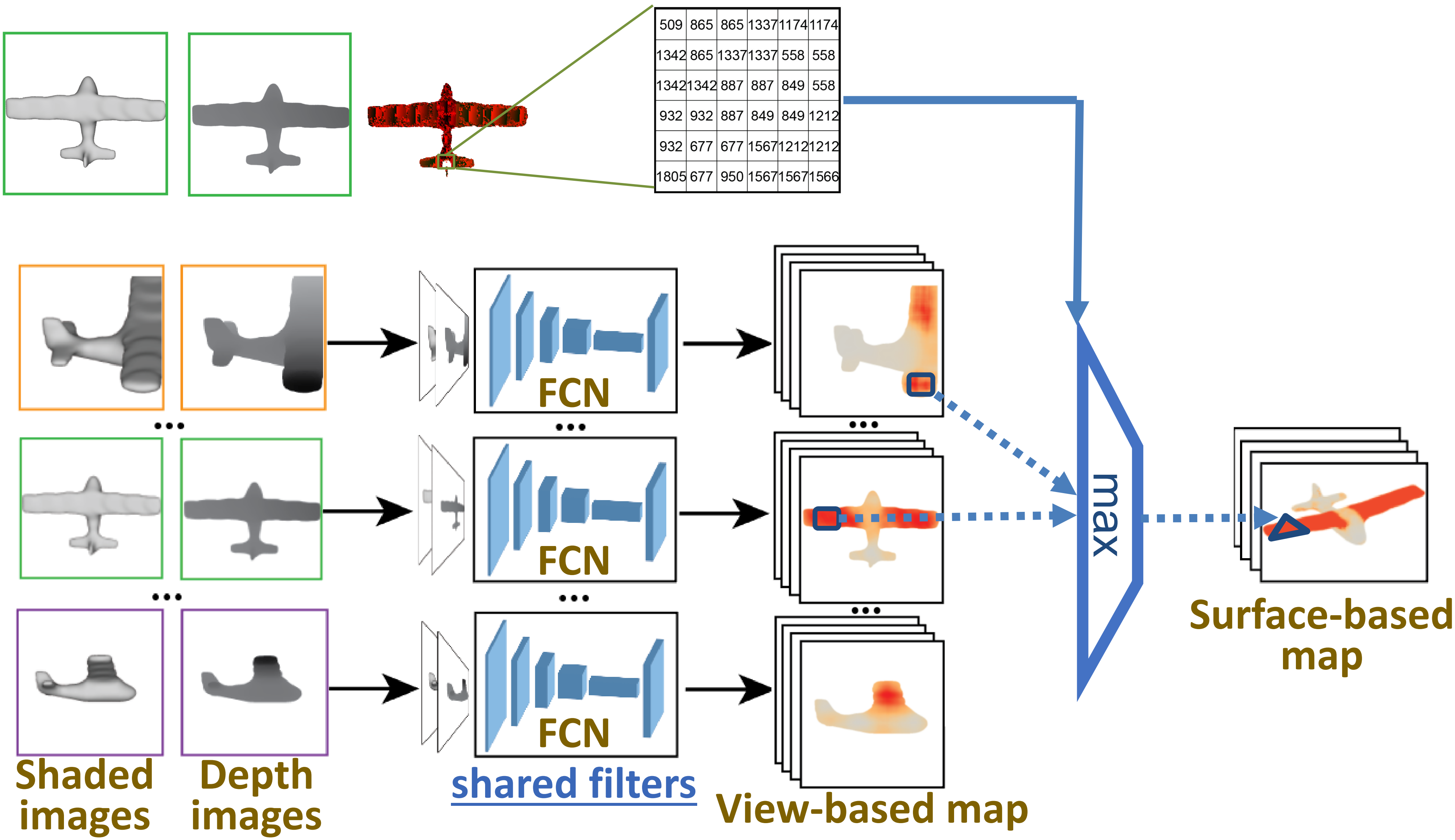


Input: shape as a collection of rendered views

For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.

Input: shape as a collection of rendered views

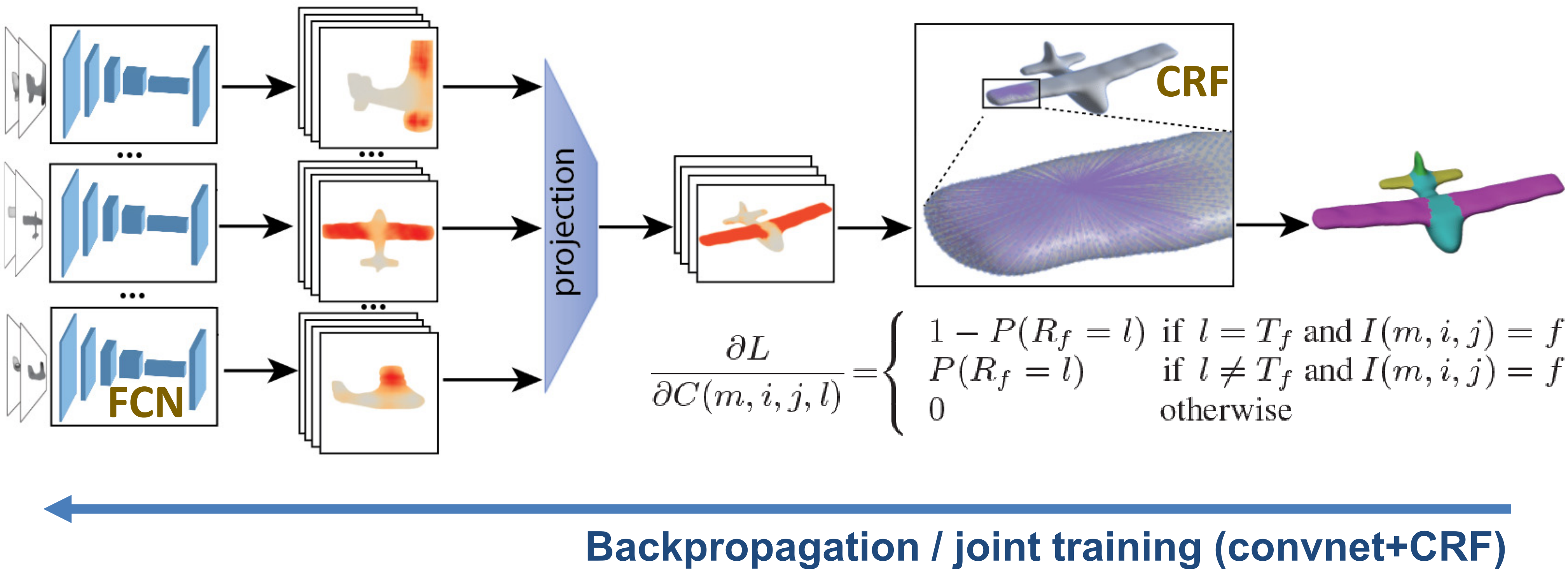
For each input shape, infer a set of viewpoints that **maximally cover its surface** across multiple distances.



509 865 865 1337 1174 1174
1342 865 1337 1337 558 558
1342 1342 887 887 849 558
932 932 887 849 849 1212
932 677 677 1567 1212 1212
1805 677 950 1567 1567 1566
FCN
FCN
FCN
max
Shaded images
Depth images
shared filters
View-based map
Surface-based map

Training

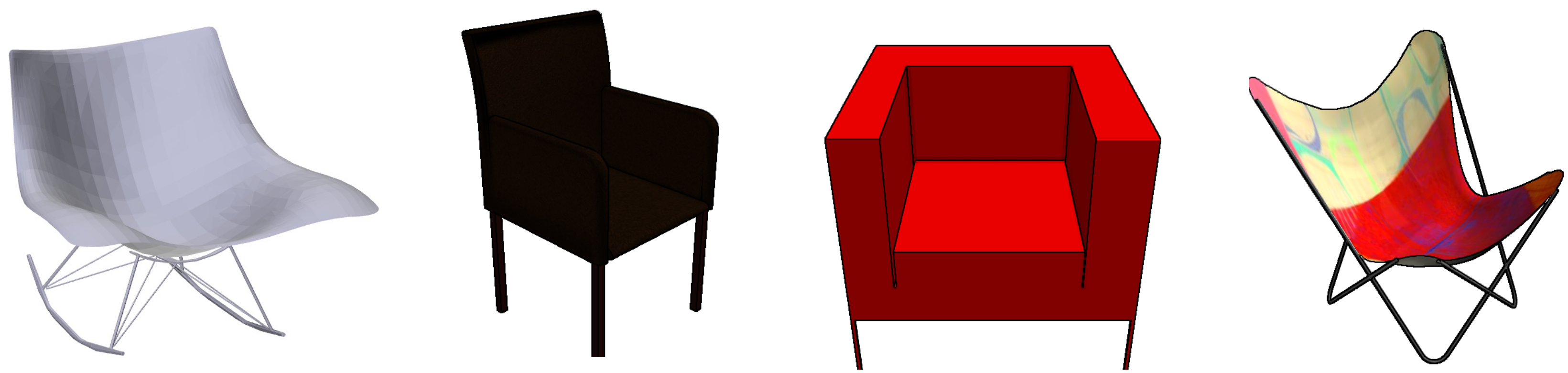
The architecture is trained **end-to-end** with analytic gradients.



$$\frac{\partial L}{\partial C(m,i,j,l)} = \begin{cases} 1 - P(R_f = l) & \text{if } l = T_f \text{ and } I(m,i,j) = f \\ P(R_f = l) & \text{if } l \neq T_f \text{ and } I(m,i,j) = f \\ 0 & \text{otherwise} \end{cases}$$

Challenges

- 3D models have **missing or non-photorealistic texture**



Challenges

- 3D models have **missing or non-photorealistic texture** (focus on **shape** instead)

ShapeNetCore: **8% improvement in labeling accuracy** for complex categories (vehicles, furniture etc)

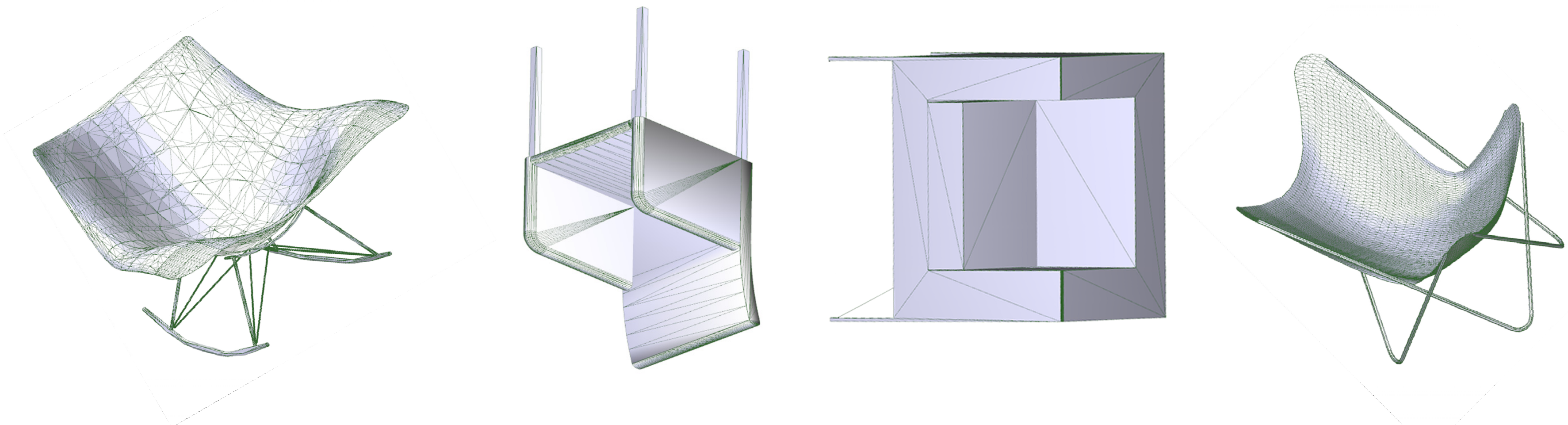
	#train/test shapes	#part labels	ShapeBoost	Guo et al.	ShapePFCN
Airplane	250 / 250	4	85.8	87.4	**90.3**
Bag	38 / 38	2	93.1	91.0	**94.6**
Cap	27 / 28	2	85.9	85.7	**94.5**
Car	250 / 250	4	79.5	80.1	**86.7**
Chair	250 / 250	4	70.1	66.8	**82.9**
Earphone	34 / 35	3	81.4	79.8	**84.9**
Guitar	250 / 250	3	89.0	89.9	**91.8**
Knife	196 / 196	2	81.2	77.1	**82.8**
Lamp	250 / 250	4	71.7	71.6	**78.0**
Laptop	222 / 223	2	86.1	82.7	**95.3**
Motorbike	101 / 101	6	77.2	80.1	**87.0**
Mug	92 / 92	2	94.9	95.1	**96.0**
Pistol	137 / 138	3	88.2	84.1	**91.5**
Rocket	33 / 33	3	79.2	76.9	**81.6**
Skateboard	76 / 76	3	91.0	89.6	**91.9**
Table	250 / 250	3	74.5	77.8	**84.8**

	fixed views	disjoint training	unary term	without pretrain.	full method
Category Avg.	87.2	87.0	83.5	86.3	**88.4**
Category Avg. (>3 labels)	83.2	82.8	78.8	82.5	**85.0**
Dataset Avg.	86.2	85.9	82.1	85.7	**87.5**
Dataset Avg. (>3 labels)	82.9	82.4	78.7	82.3	**84.7**

Table 3. Labeling accuracy on ShapeNetCore for degraded variants of our method.

Key Observations

3D models have **arbitrary orientation** "in the wild".



Consistent shape orientation only in specific, well-engineered datasets (often with manual intervention, no perfect alignment algorithm)

Comparisons pitfalls

- Method A assumes **consistent shape alignment** (or upright orientation), method B doesn't.
  - Well, you may get much better numbers for B by changing its input!
- Convnet A has **orders of magnitude more parameters** than Convnet B.
  - It might also be easy to get better numbers for B, if you increase its number of filters!
- Convnet A has an **architectural "trick"** that Convnet B could also have (e.g., U-net, ensemble).
  - Why not apply the same trick to B?
- Methods are largely tested on training data because of **duplicate or near-duplicate shapes**.
  - **The more you overfit, the better! Ouch!**
  - 3DShapeNet has many identical models, or models with tiny differences (e.g., same airplane with different rockets)...