# ODIN: A Single Model for 2D and 3D Segmentation

Stefano Ruggiero

# Why this paper ?

- Very recent : last update 25 Jun 2024
- Interesting idea

# The autors

Ayush Jain[1], Pushkal Katara[1], Nikolaos Gkanatsios[1], Adam W. Harley[2],
Gabriel Sarch[1], Kriti Aggarwal[3], Vishrav Chaudhary[3], Katerina Fragkiadaki[1]
[1]Carnegie Mellon University, [2]Stanford University, [3] Microsoft
{ayushj2, pkatara ngkanats,gsarch,kfragki2}@andrew.cmu.edu
aharley@cs.stanford.edu, {kragga,vchaudhary}@microsoft.com

# Things to know (before starting)

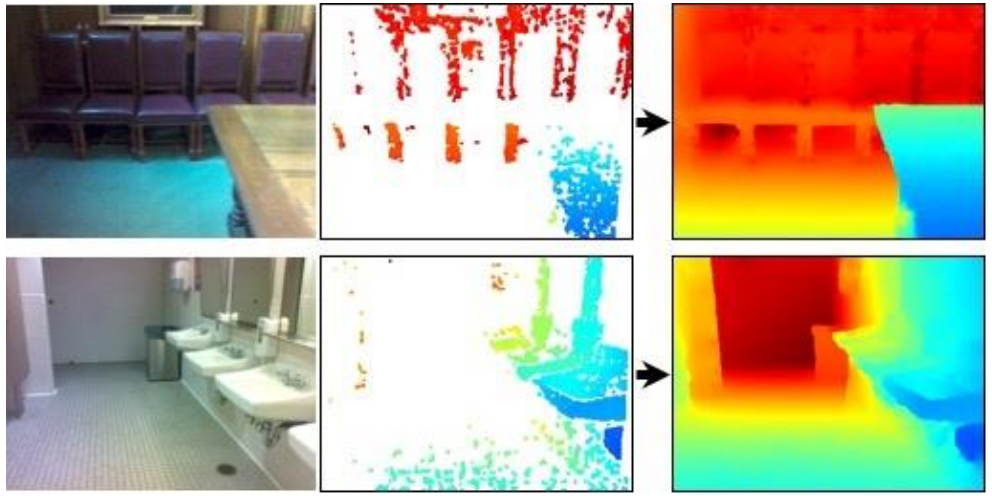
1) RGB-D images
2) Voxelization
3) Embodied AI
4) Ablation AI
5) Joint Training
6) Trilinear interpolation (EXTRA)
7) ScanNet
8) ScanNet200
9) Matterport3D
10) S3DIS
11) ResNet50
12) Swin Transformer
13) Mask2Former
14) Mask3D
15) Skip connection
16) Dot product
17) AI2THOR
18) TEACh e ALFRED

# Things to know (before starting)

RGB-D image

Is a combination of a RGB image and its corresponding depth image, better known as depth map.

[https://communities.springernature.com/posts/depth-in-focus-d-component-of-rgb-d-images-and-videos]



Input: RGB-D from sensor Complete Depth

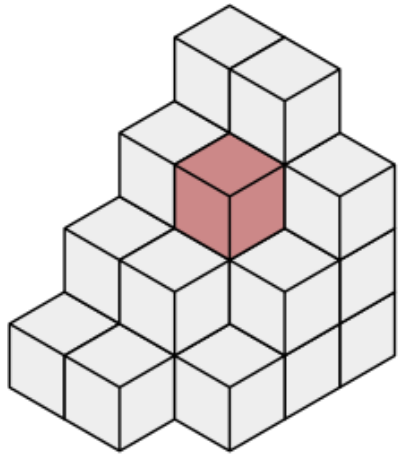
[https://deepcompletion.cs.princeton.edu/]

# Things to know (before starting)

Voxelization

A voxel is a three-dimensional counterpart to a pixel. It represents a value on a regular grid in a three-dimensional space.

[https://en.wikipedia.org/wiki/Voxel]



A set of voxels in a stack, with a single voxel shaded

# Things to know (before starting)

Embodied AI

Refers to artificial intelligence systems that can interact with and learn from their environments using a suite of technologies that include sensors, motors, machine learning and natural language processing.

[https://www.techtarget.com/searchenterpriseai/definition/embodied-AI]

# Things to know (before starting)

Ablation AI

Ablation is the removal of a component of an AI system. An ablation study aims to determine the contribution of a component to an AI system by removing the component, and then analyzing the resultant performance of the system.

[https://en.wikipedia.org/wiki/Ablation_(artificial_intelligence)]

# Things to know (before starting)

Joint Training

Advanced machine learning technique where multiple models learn together on a shared dataset.

[https://www.linkedin.com/pulse/what-joint-training-aionlinecourse-j1iac]

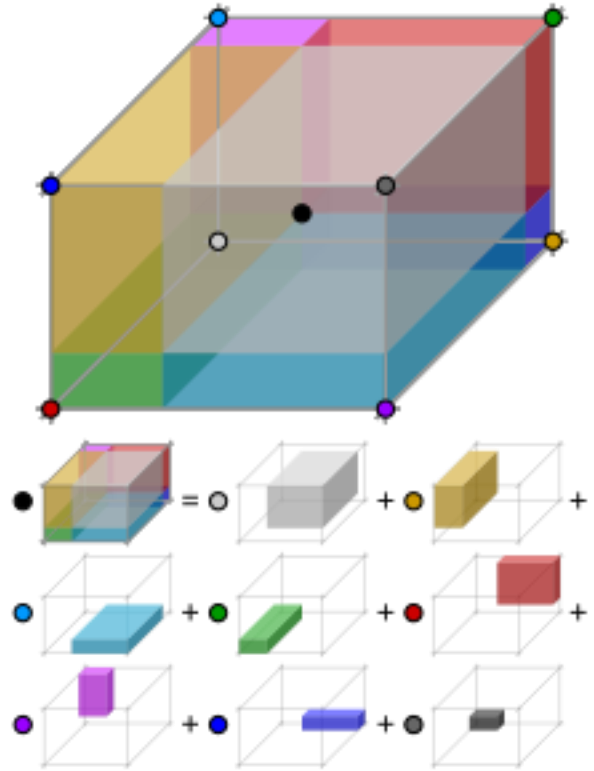
# Things to know (before starting)

Trilinear interpolation (EXTRA)

Is a method of multivariate interpolation on a 3-dimensional regular grid.

It approximates the value of a function at an intermediate point (x,y,z) within the local axial rectangular prism linearly.

[https://en.wikipedia.org/wiki/Trilinear_interpolation]



A geometric visualisation of trilinear interpolation. The product of the value at the desired point and the entire volume is equal to the sum of the products of the value at each corner and the partial volume diagonally opposite the corner.

# Things to know (before starting)

ScanNet

Is an RGB-D video dataset containing 2.5 million views in more than 1500 scans, annotated with 3D camera poses, surface reconstructions, and instance-level semantic segmentations. 20 classes.

[https://github.com/ScanNet/ScanNet]

# Things to know (before starting)

ScanNet200

Is a benchmark which studies 200-class 3D semantic segmentation - an order of magnitude more class categories than previous 3D scene understanding benchmarks.

[https://rozdavid.github.io/scannet200]

# Things to know (before starting)

ResNet50

Is a deep convolutional neural network (CNN) architecture that was developed by Microsoft Research in 2015. It is a variant of the popular ResNet architecture, which stands for “Residual Network.” The “50” in the name refers to the number of layers in the network, which is 50 layers deep.

[https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f]

# Things to know (before starting)

Swin Transformer

Is an hierarchical Transformer serves as a general-purpose backbone for computer vision, whose representation is computed with Shifted windows.

[https://arxiv.org/abs/2103.14030]

# Things to know (before starting)

Mask2Former

Masked-attention Mask Transformer, key components include masked attention, which extracts localized features by constraining cross-attention within predicted mask regions.

[https://arxiv.org/pdf/2112.01527]

# Things to know (before starting)

Mask3D

A Transformer-based approach for 3D semantic instance segmentation, each object instance is represented as an instance query.

[https://jonasschult.github.io/Mask3D/]

# Things to know (before starting)

Skip connection

Skips some of the layers in the neural network and feeds the output of one layer as the input to the next layers.

[https://www.analyticsvidhya.com/blog/2021/08/all-you-need-to-know-about-skip-connections/#h-why-skip-connections]

# Things to know (before starting)

Dot product

Measures the similarity between two vectors in feature space

[https://bowenc0221.github.io/maskformer/]

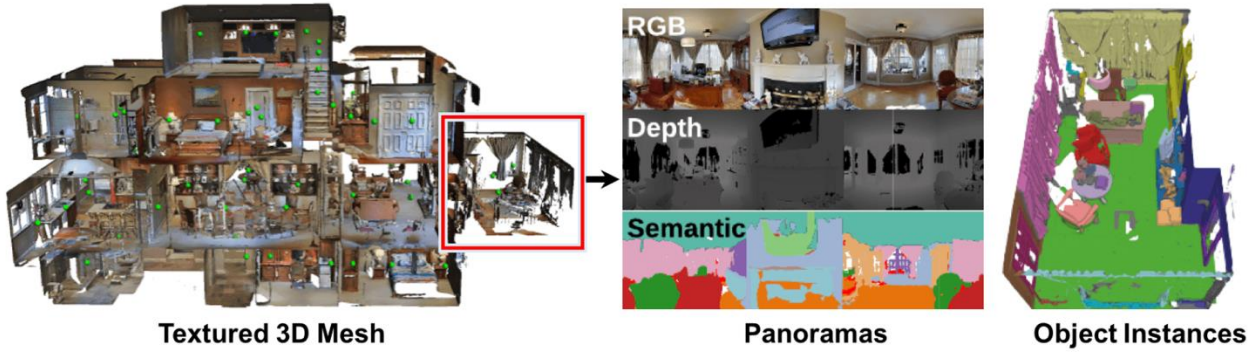
# Things to know (before starting)

Matterport3D

A large-scale RGB-D dataset containing 10,800 panoramic views from 194,400 RGB-D images of 90 building-scale scenes.

[https://niessner.github.io/Matterport/]

The official benchmark of Matterport3D tests on 21 classes; however, OpenScene also evaluates on 160 classes to compare with state-of-theart models on long-tail distributions.

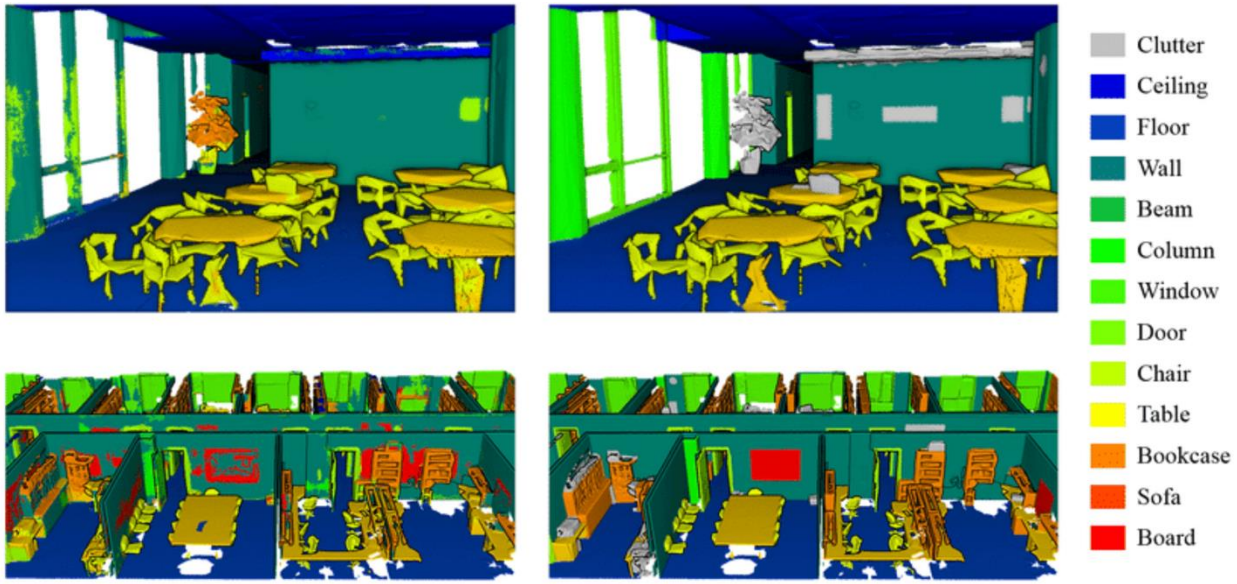


# Things to know (before starting)

## S3DIS

The Stanford 3D Indoor Scene Dataset (**S3DIS**) dataset contains 6 large-scale indoor areas with 271 rooms. Each point in the scene point cloud is annotated with one of the 13 semantic categories.

[https://paperswithcode.com/dataset/s3dis]

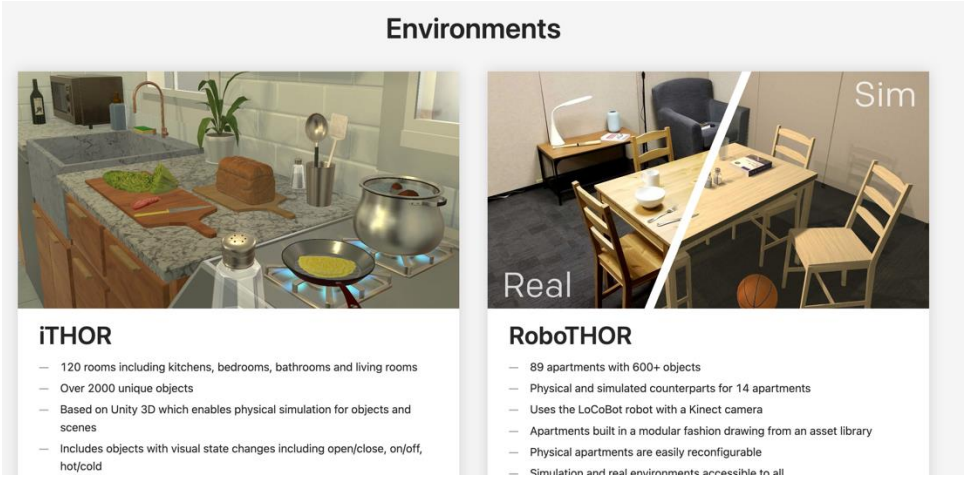


# Things to know (before starting)

AI2THOR

AI2-Thor is an interactive environment for embodied AI.

[https://paperswithcode.com/dataset/ai2-thor]

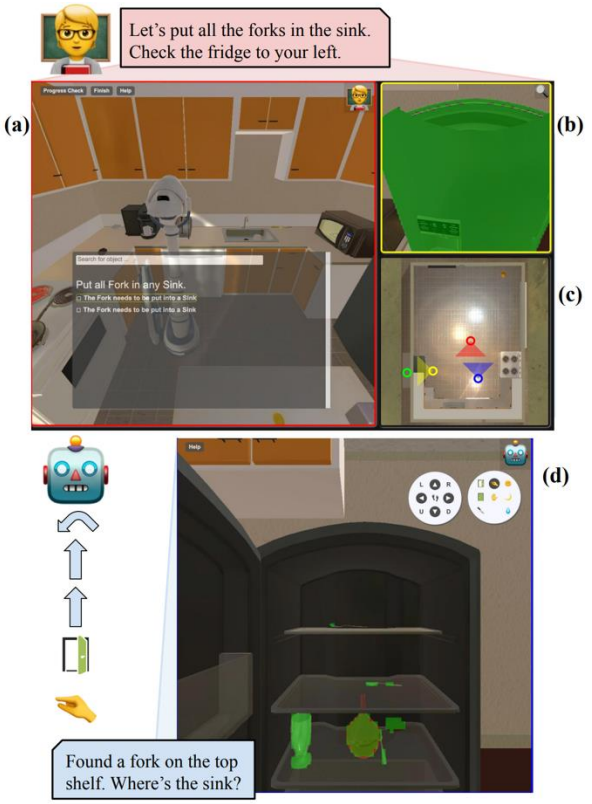


# Things to know (before starting)

## TEACh e ALFRED

Embodied ai simulator (Task-driven Embodied Agents that Chat)

[https://arxiv.org/pdf/2110.00534]

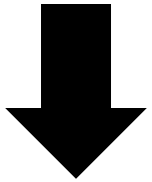


# Introduction to ODIN

# Introduction to ODIN

- There has been a surge of interest in porting 2D foundational image features to 3D scene understanding.

- State-of-the-art on established 3D segmentation benchmarks (such as ScanNet and ScanNet200) still consists of models that operate directly in 3D, without any 2D pre-training stage.

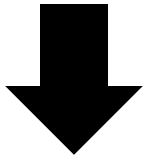


Why is it so difficult to yield improvements in these 3D tasks?

# Introduction to ODIN

Why is it so difficult to yield improvements in these 3D tasks?

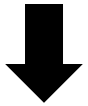
- Part of the issue lies in a key implementation detail underlying 3D benchmark evaluations.
- Benchmarks like ScanNet do not actually ask methods to use RGB-D images as input, even though this is the sensor data: first register all RGBD frames into a single colored point cloud and reconstruct the scene as cleanly as possible, relying on manually tuned stages for bundle adjustment.



inconsistent pipeline

# Introduction to ODIN

Why is it so difficult to yield improvements in these 3D tasks?



inconsistent pipeline

- This pipeline is inconsistent with the goals of embodied vision (and typical 2D vision), which involves dealing with actual sensor data and accounting for missing or partial observations.

# Introduction to ODIN

Hypothesis: «We therefore hypothesize that method rankings will change, and the impact of 2D pre-training will become evident »

- ScanNet undergoes complex mesh reconstruction steps, introducing high misalignments.
- Misalignments introduced can cause methods processing sensor data directly to underperform compared to those trained and tested on provided point clouds.
- Some datasets lack access to raw RGB-D data.
- While mesh reconstruction has its applications, many realtime applications need to directly process sensor data.

# Introduction to ODIN

- Problems:

1. Inconsistent pipeline for embodied ai and real time operation
2. «method rankings will change»

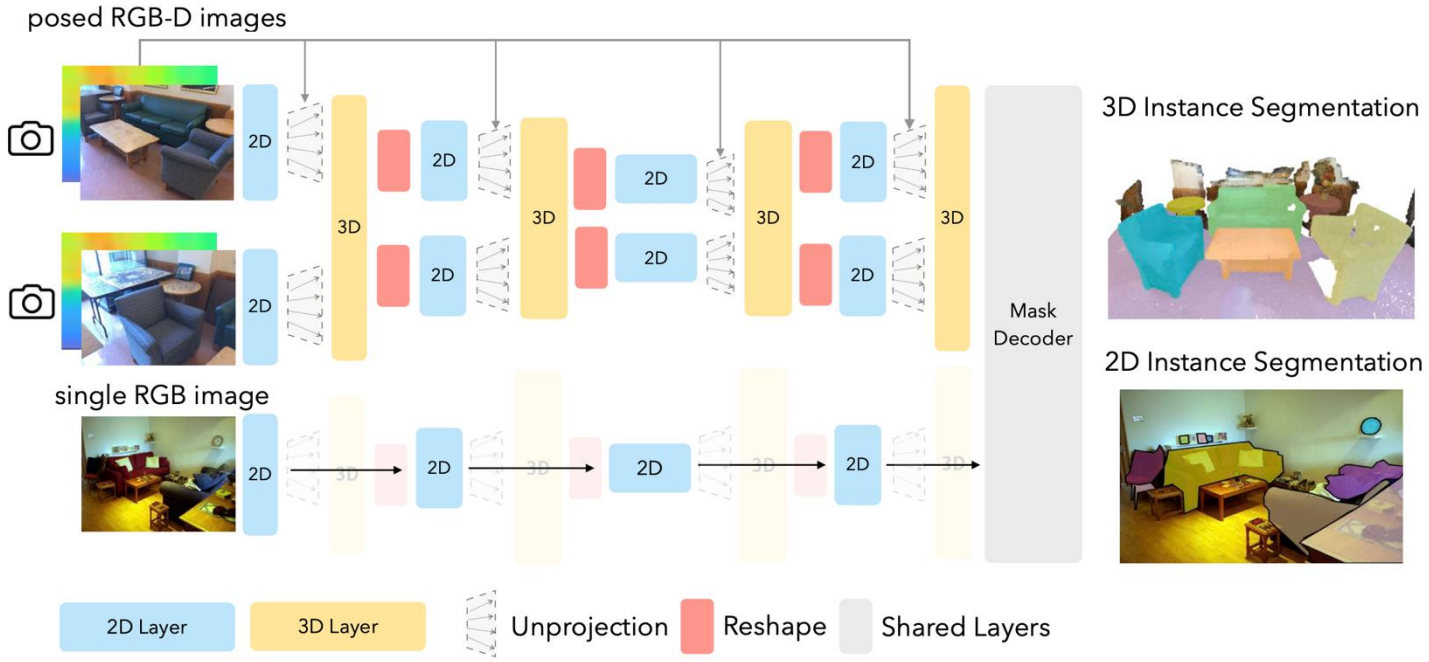
- Solution:

Omni-Dimensional INstance segmentation (ODIN), a model for 2D and 3D object segmentation and labelling that can parse single-view RGB images and/or multiview posed RGB-D images.

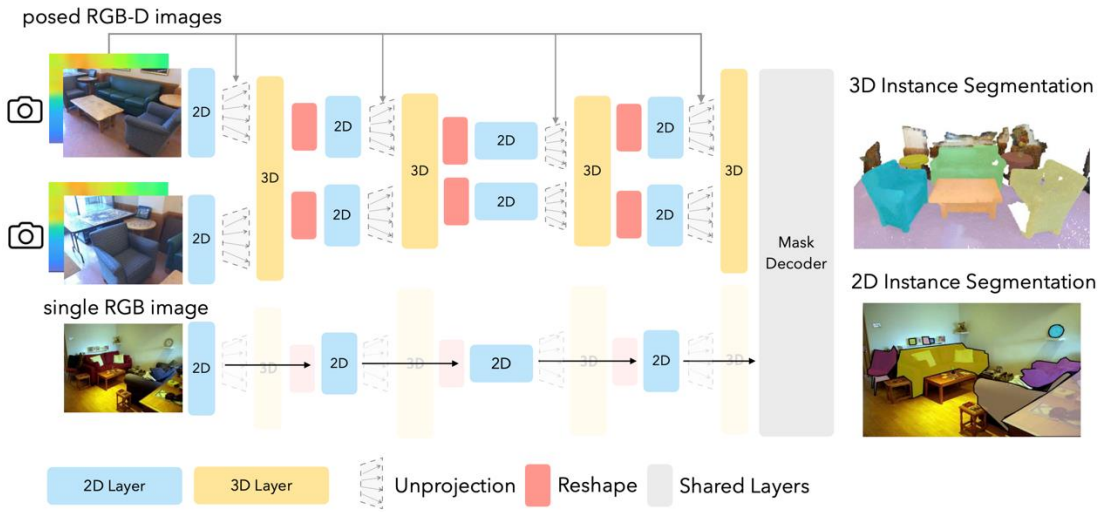
# ODIN architecture

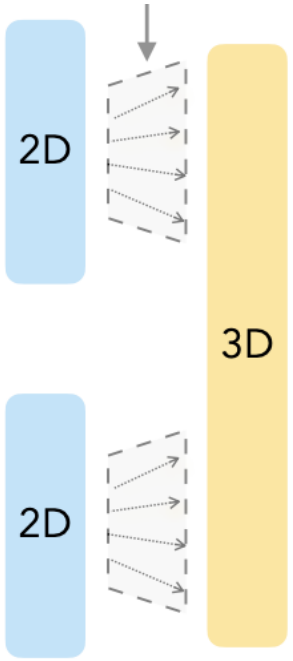
# ODIN architecture

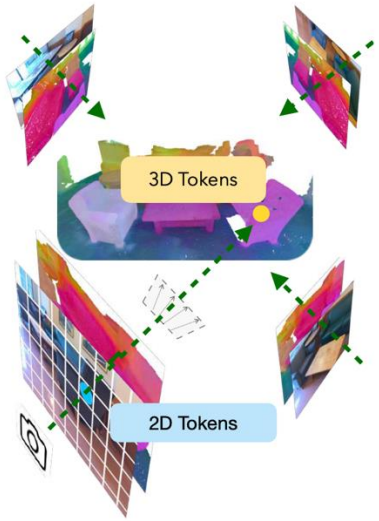
ODIN architecture instantiates a single unified U-Net which interleaves 2D and 3D layers and can handle both 2D and 3D perception tasks with a single unified architecture.



# ODIN architecture

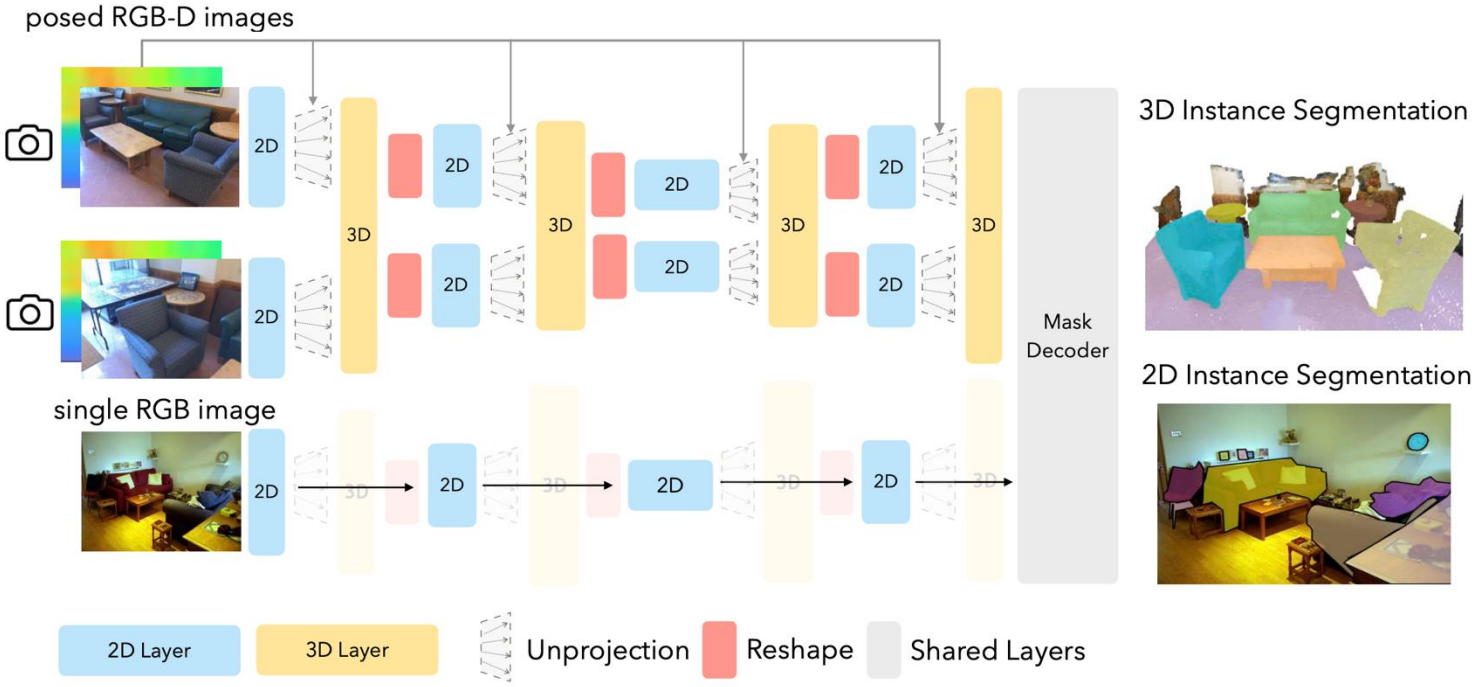


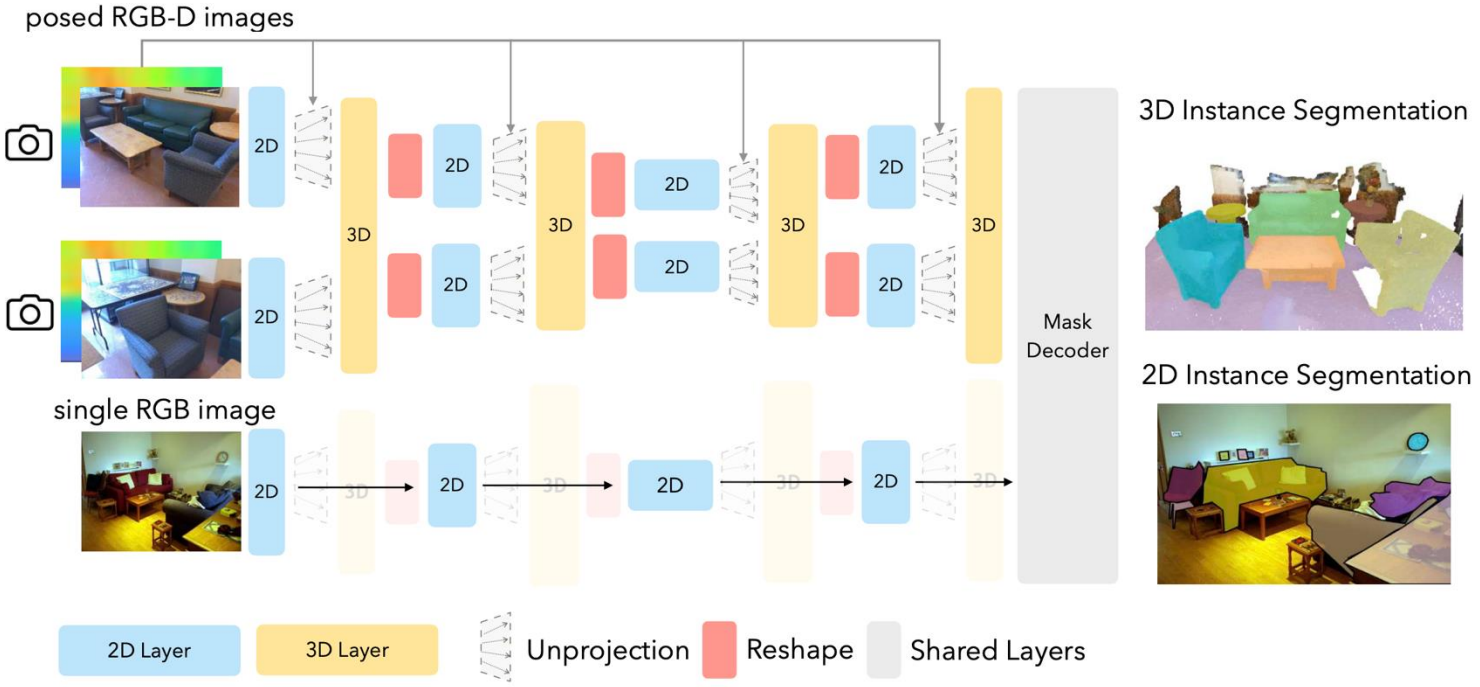

At each 2D-to-3D transition, it unprojects 2D tokens to their 3D locations using the depth maps and camera parameters, and at each 3D-to-2D transition, it projects 3D tokens back to their image locations.

# ODIN architecture




This model differentiates between 2D and 3D features through the positional encodings of the tokens involved, which capture pixel coordinates for 2D patch tokens and 3D coordinates for 3D feature tokens.

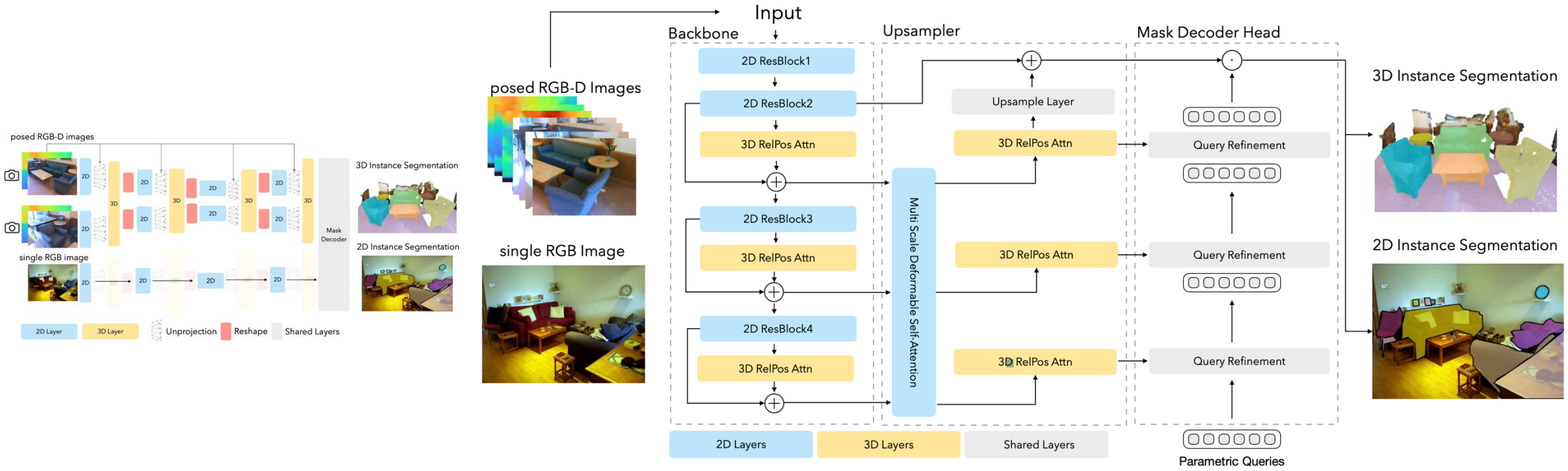
# ODIN architecture




Given a posed RGB-D sequence as input, ODIN alternates between a **within-view** 2D fusion and a **cross-view** 3D fusion.

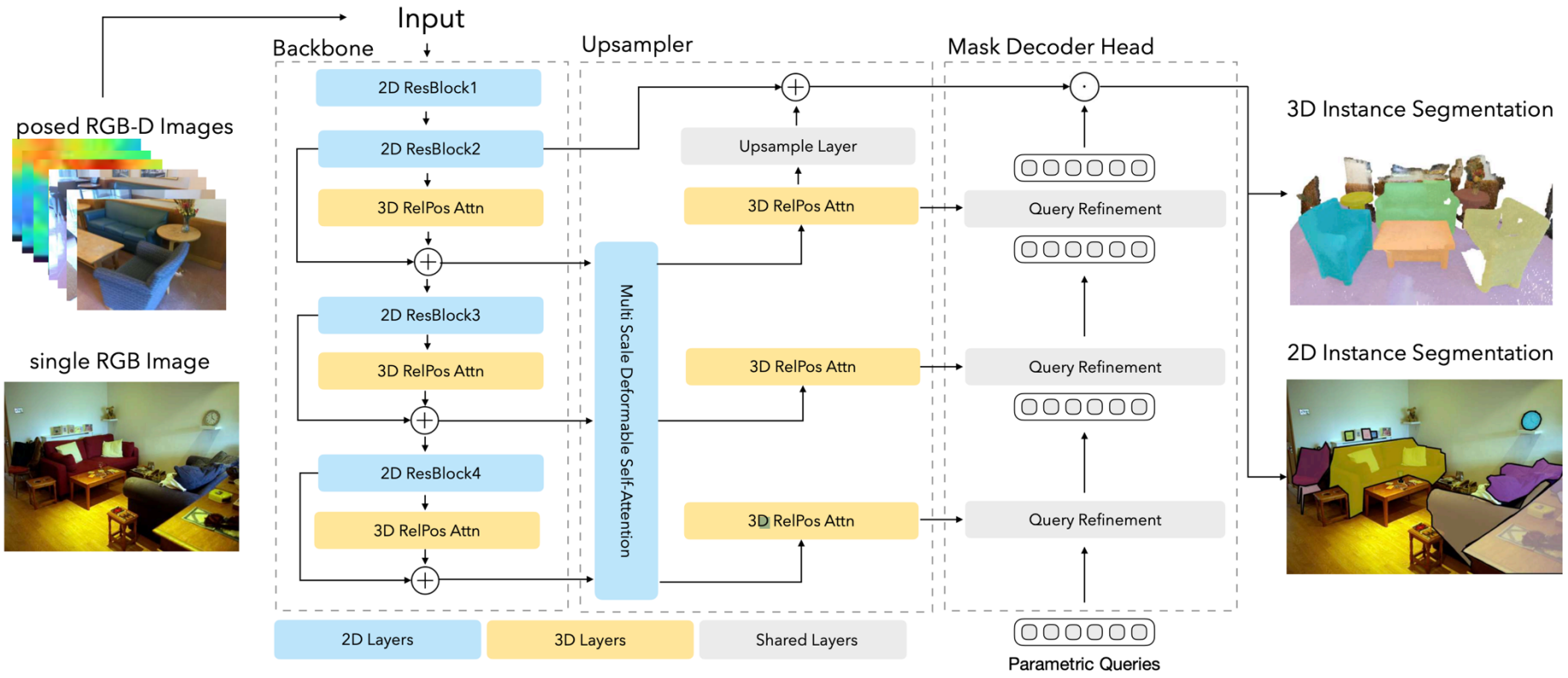
When the input is a single RGB image, the 3D fusion layers are skipped.

# ODIN architecture – first look



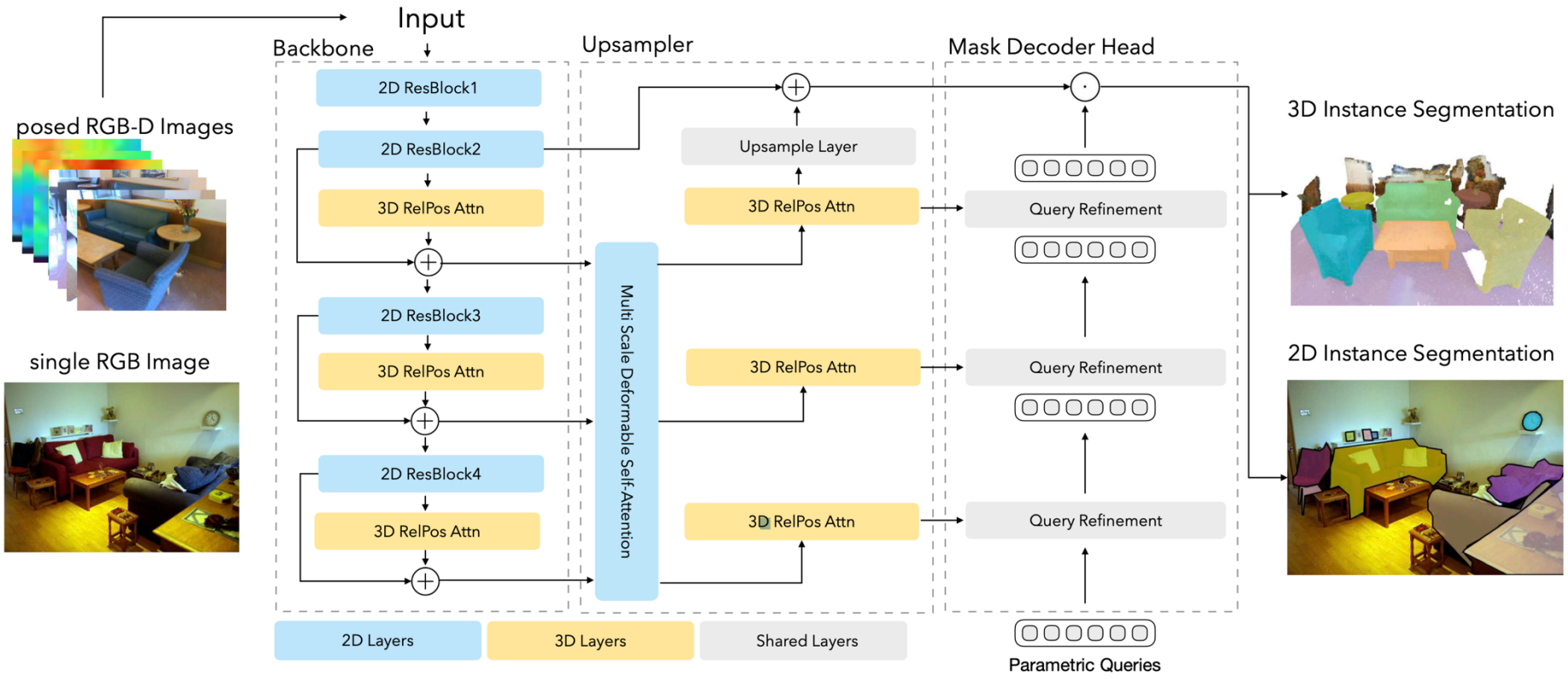

Example: take either a single RGB image or a set of posed RGB-D images and outputs the corresponding 2D or 3D instance segmentation masks and their semantic labels.

# ODIN architecture – first look



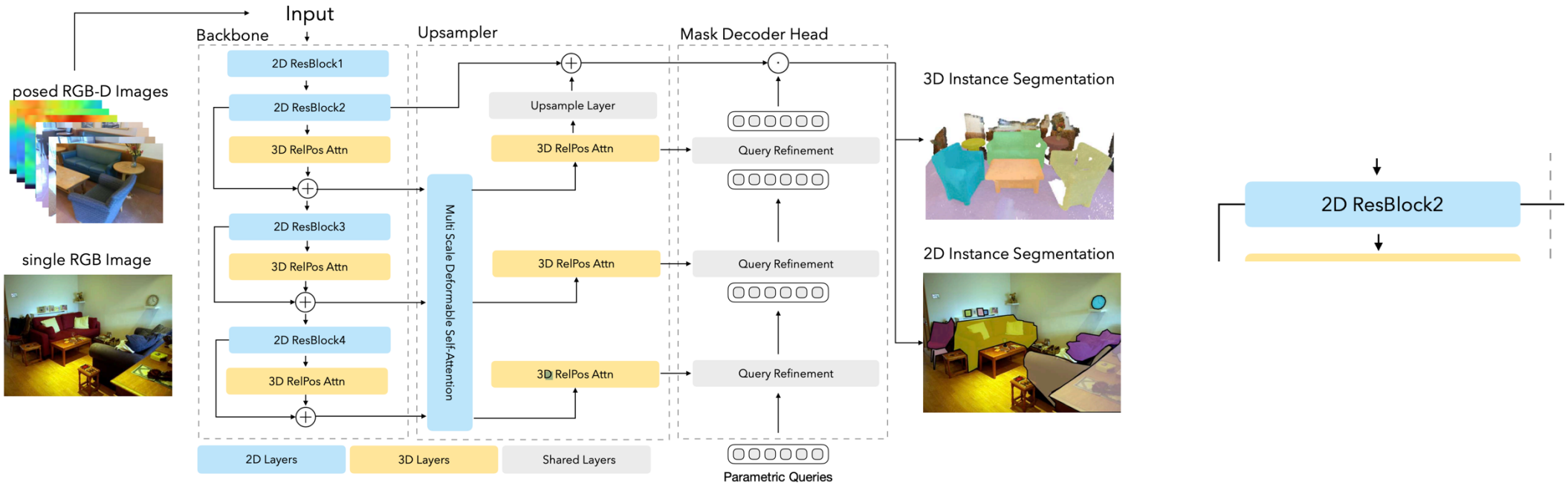

ODIN alternates between a 2D within-view fusion and a 3D attention-based cross-view fusion, as illustrated in blue blocks and yellow blocks.

# ODIN architecture – first look




Notably, ODIN shares the majority of its parameters across both RGB and multiview RGB-D inputs → enables the use of pre-trained 2D backbones.

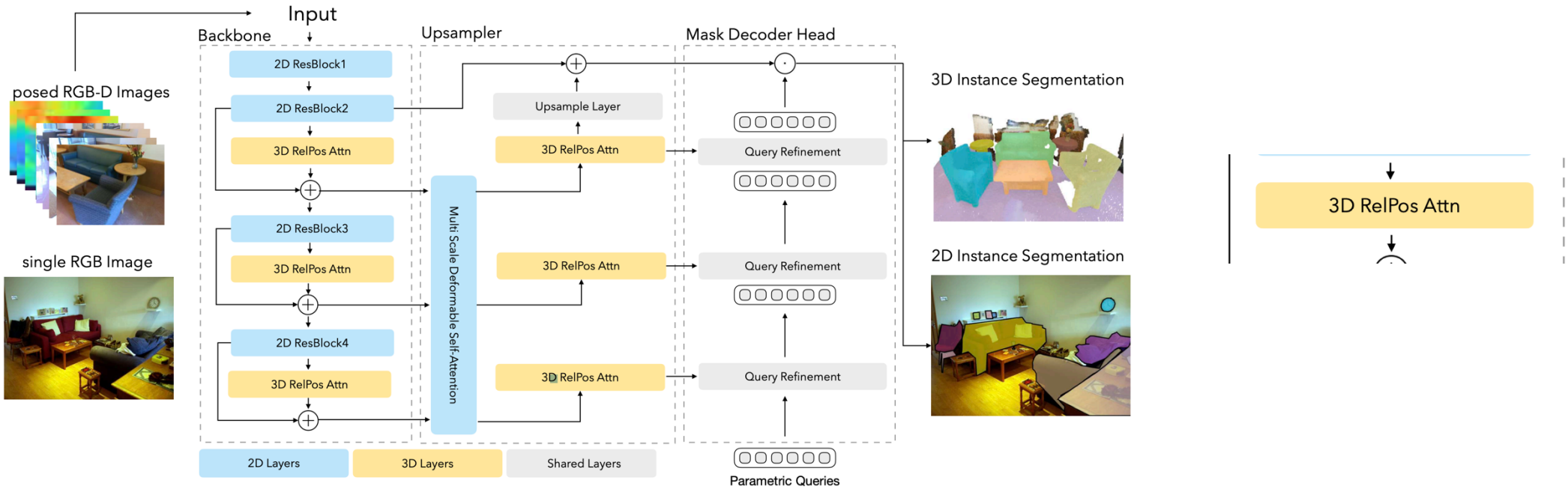
# ODIN architecture – first look




Within-view 2D fusion:

ODIN starts from a 2D backbone (such as ResNet50 or Swin Transformer) pre-trained for 2D COCO instance segmentation to extract feature maps. (following Mask2Former )

# ODIN architecture – first look



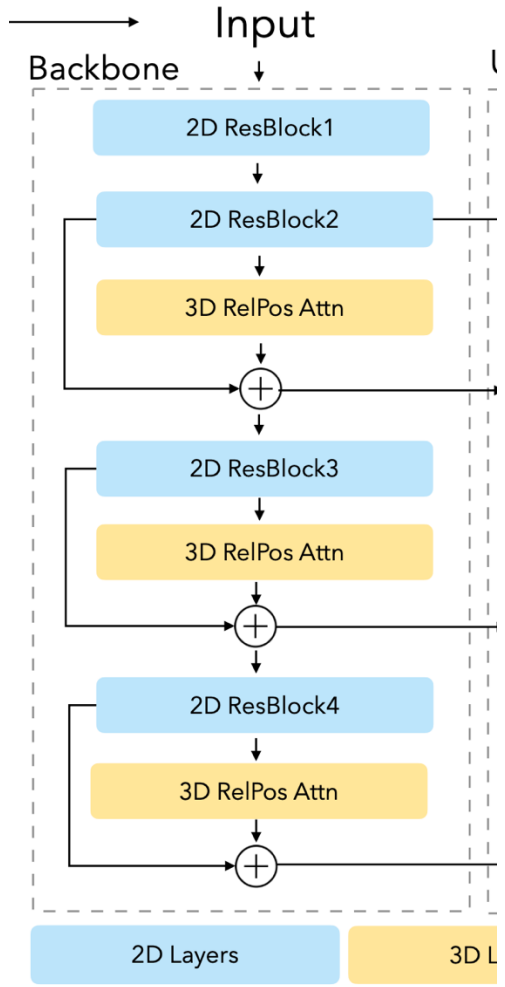

Cross-view 3D fusion:

The goal of cross-view fusion is to make the individual images' representations consistent across views. (demonstrated in ablations section)

# ODIN architecture – first look

Why interleaving within-view and cross-view contextualization ?

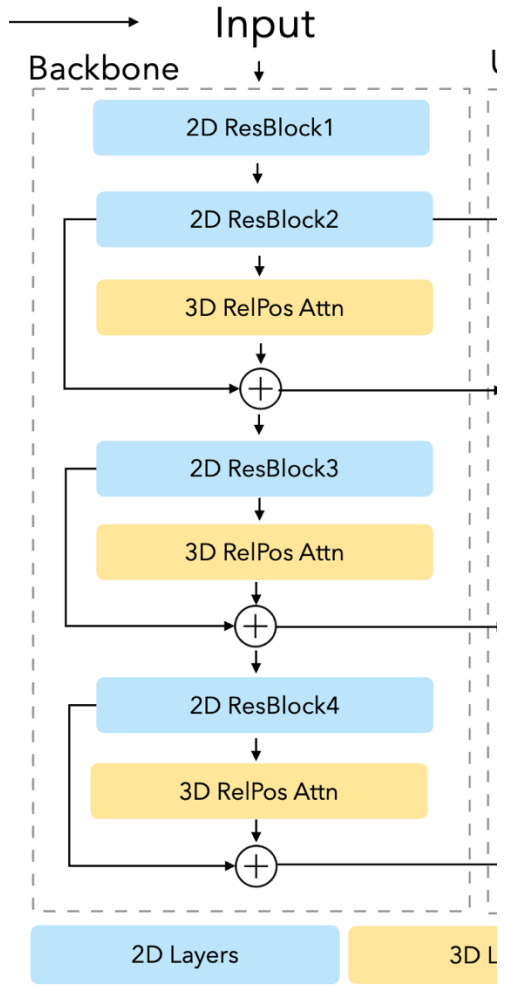
To utilize the pre-trained features from the 2D backbone while also fusing features across views, making them 3D-consistent.



# ODIN architecture – first look

**Backbone**

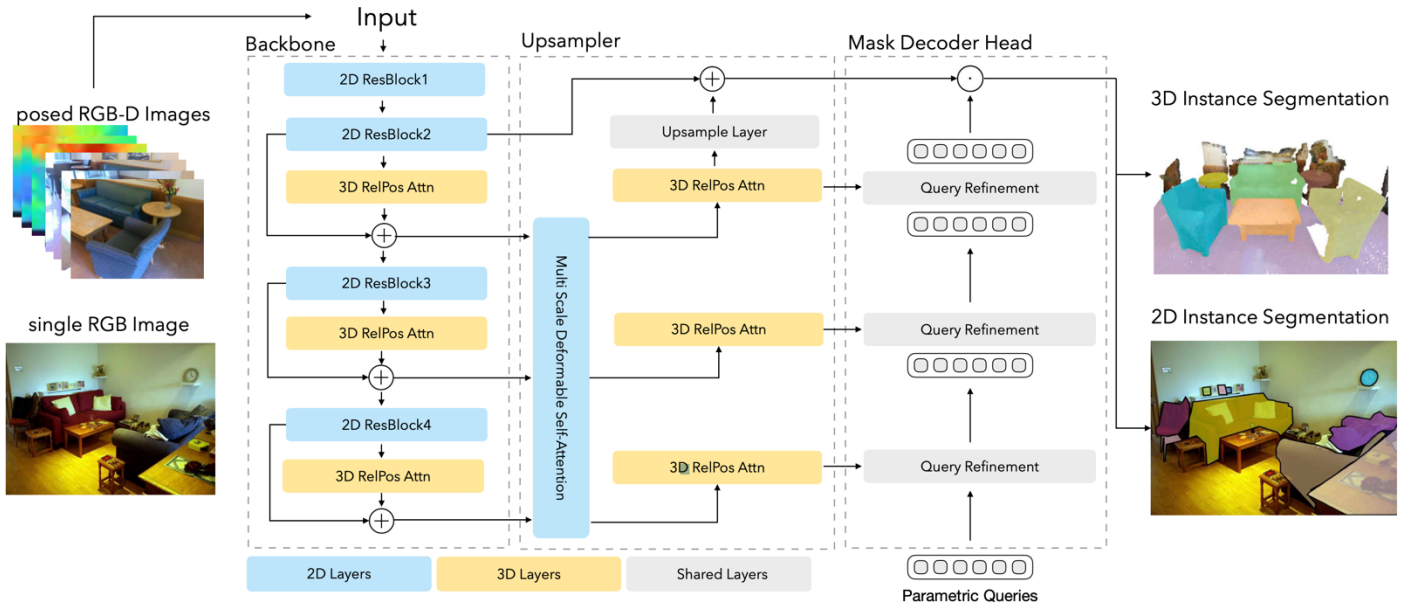
Inputs are interleaved in 2D within-view fusion layers and 3D cross-view attention layers to extract feature maps of different resolutions (scales).

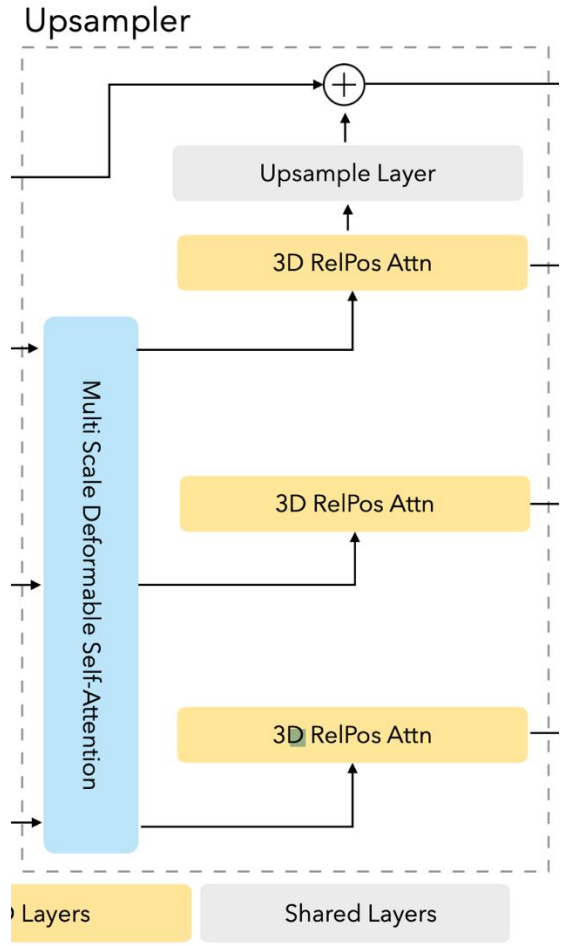


# ODIN architecture – first look

**Upsampler**

These feature maps exchange information through a multi-scale attention operation (→ Multi Scale Deformable Self-Attention). Additional 3D fusion layers are used to improve multiview consistency.

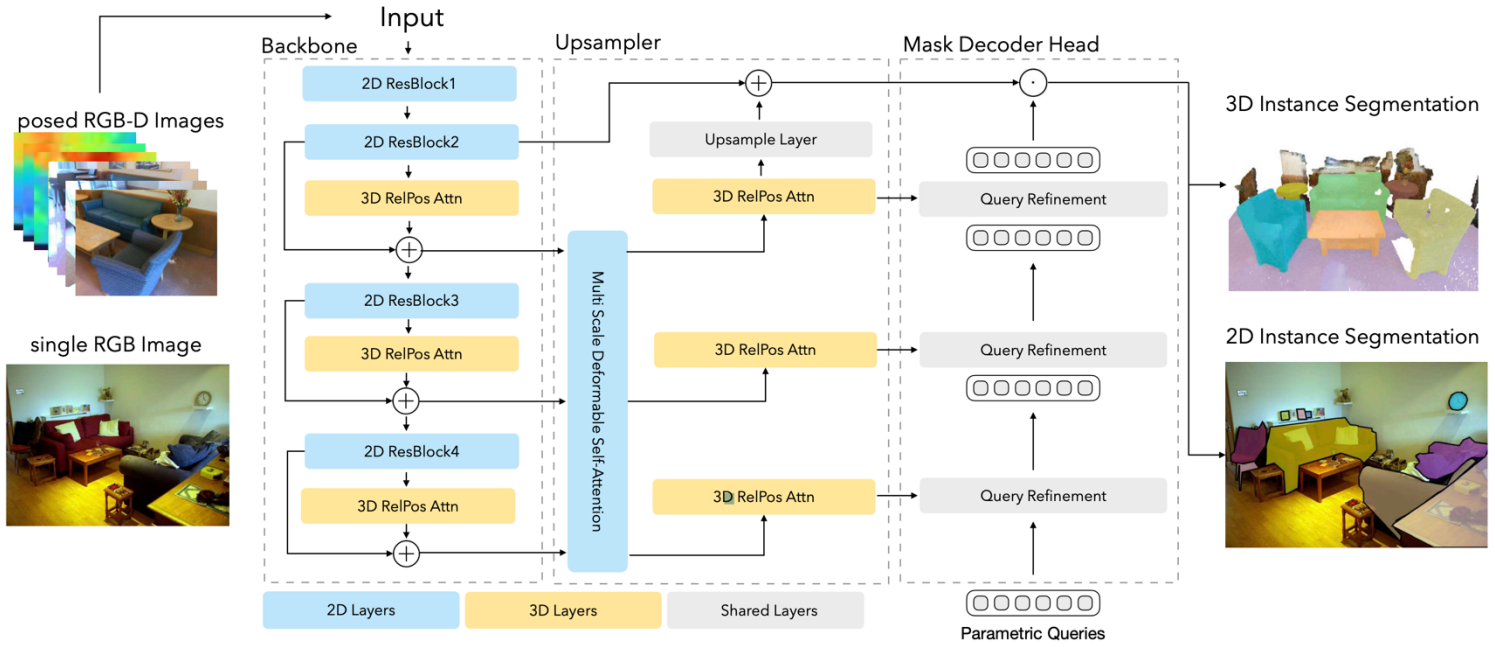


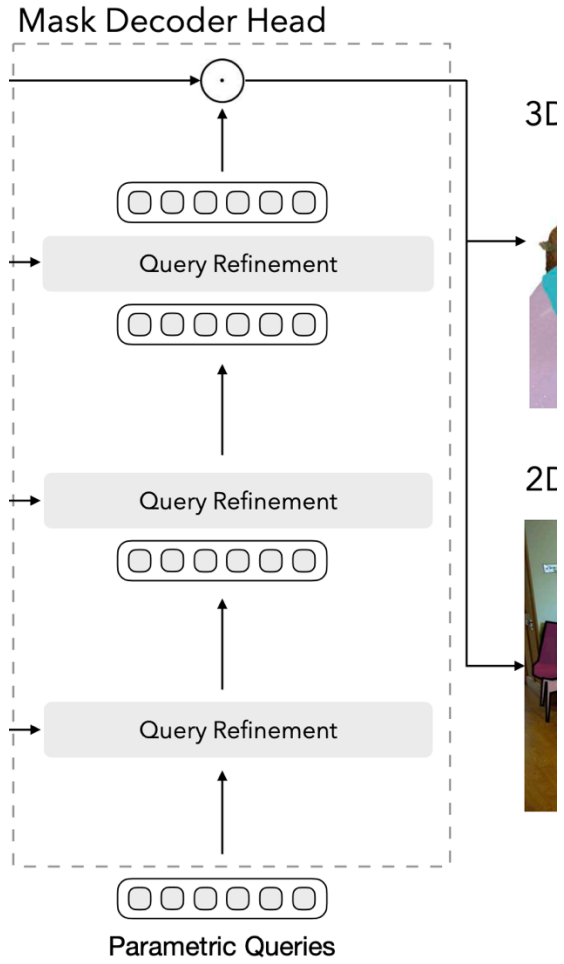



# ODIN architecture – first look

**Decoder**

A mask decoder head is used to initialize and refine learnable slots that attend to the multi-scale feature maps and predict object segments (masks and semantic classes).



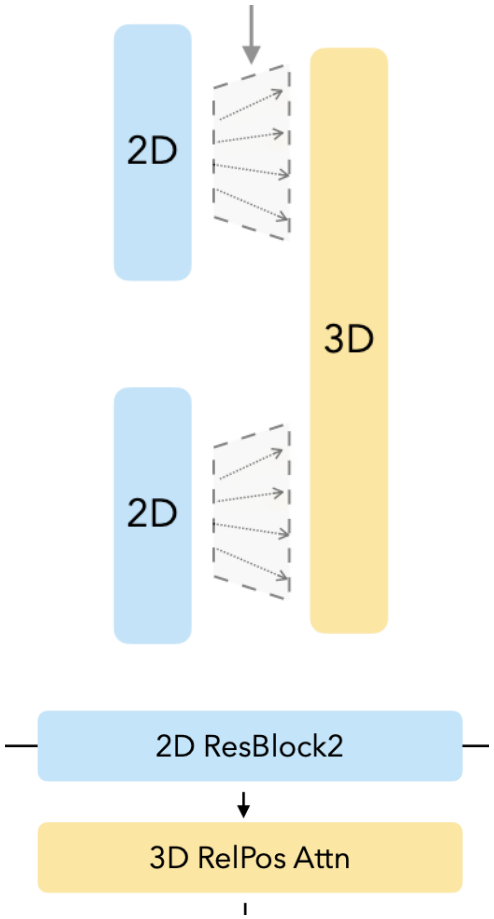



# ODIN architecture – deeper look

**2D-to-3D Unprojection**

ODIN unprojects each 2D feature map to 3D by lifting each feature vector to a corresponding 3D location, using nearest neighbor depth and known camera intrinsic and extrinsic parameters.

The resulting featurized point cloud undergoes voxelization, where the 3D space is discretized into a volumetric grid.



# ODIN architecture – deeper look

**3D k-NN Transformer with Relative Positions**

3D RelPos Attn

ODIN fuses information across 3D tokens using k-nearest-neighbor attention with relative 3D positional embeddings.

Each 3D token attends to its k nearest neighbors.

The positional embeddings in this operation are relative to the query token’s location.

# ODIN architecture – deeper look

**3D k-NN Transformer with Relative Positions**

ODIN achieves this by encoding the distance vector between a token and its neighbour with an MLP (feedforward neural network).

The positional embedding for the query is simply encoding of the 0 vector:

$$\text{query}_{\text{pos}} = \text{MLP}(0);$$

This indicates that the query does not have a specific relative position associated with it, but a “neutral” representation

$$\text{key}_{\text{pos}} = \text{MLP}(p_i - p_j),$$

The relative position between two points is calculated as a difference

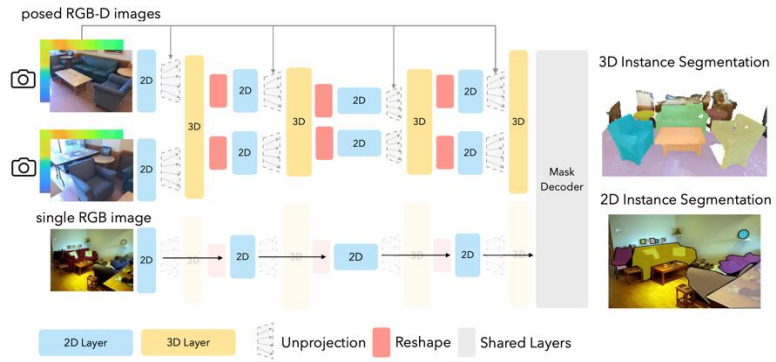
where pi represents the 3D tokens, shaped N × 1 × 3, and pj represents the k nearest neighbors of each pi , shaped N×k×3.

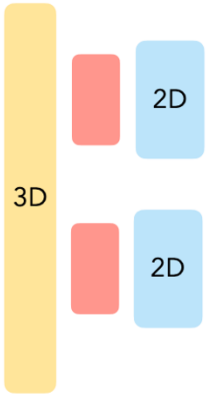
# ODIN architecture – deeper look

## 3D-to-2D Projection

Projects the features back to their original 2D locations.

First copy the feature of each voxel to all points within that voxel, then reshape these points back into multiview 2D feature maps, so that they may be processed by the next 2D module.



# ODIN architecture – deeper look

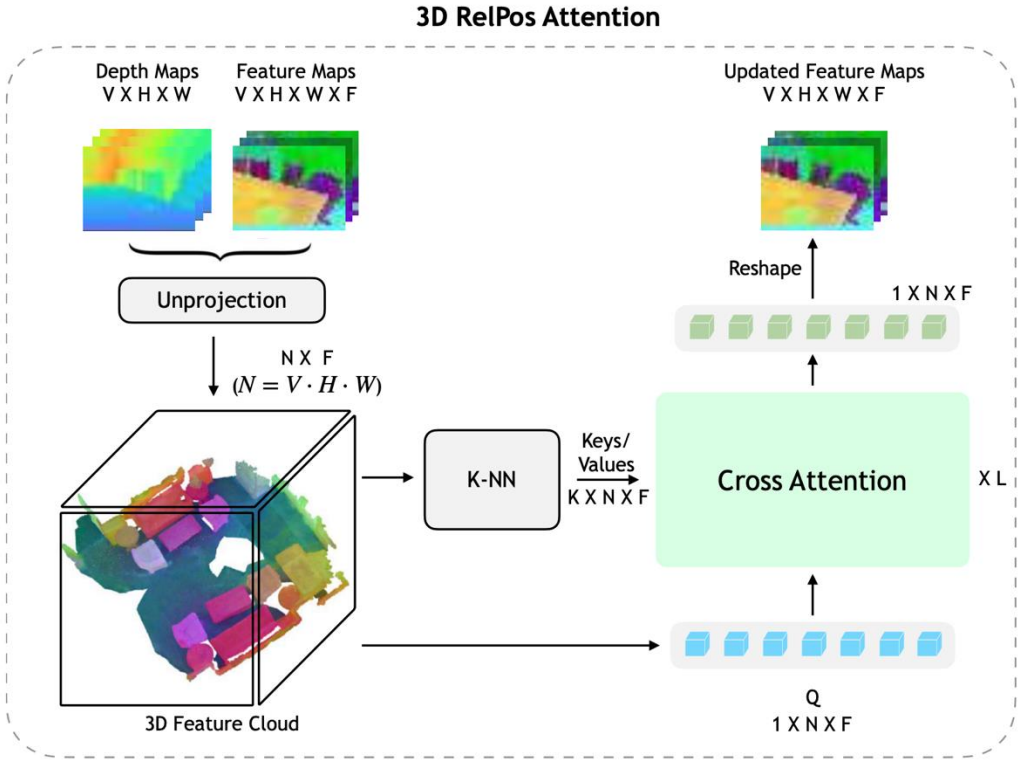
## The complete input pipeline

1. 3D RelPos Attention module takes as input the depth, camera parameters and feature maps from all views, gets 3D tokens. Each 3D token serves as a query.

2. The K-Nearest Neighbors of each 3D token become the corresponding keys and values.

3. The 3D tokens attend to their neighbours for L layers and update themselves.

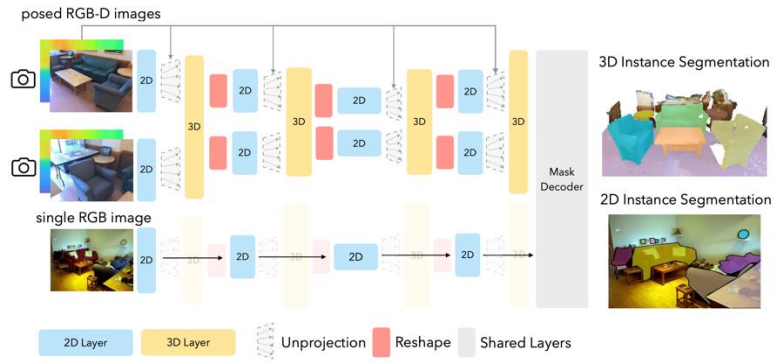
4. Finally, the 3D tokens are mapped back to the 2D feature map by simply reshaping the 3D feature cloud to 2D multi-view feature maps.

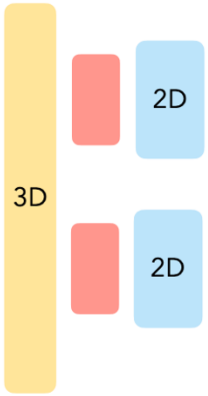


# ODIN architecture – deeper look

## 3D-to-2D Projection

Note: The features vectors are unchanged in this transition; the difference lies in their interpretation and shape.
In 2D the features are shaped V × H × W × F, representing a feature map for each viewpoint, and in 3D they are shaped N×F, representing a unified feature cloud, where N = V · H · W.



# ODIN architecture – deeper look

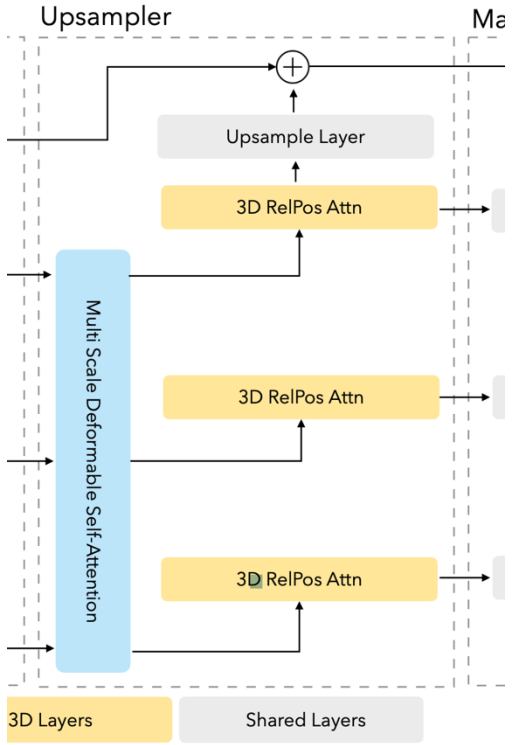
## Cross-scale fusion and upsampling

After multiple single-view and cross-view stages, ODIN has access to multiple features maps per image, at different resolutions.

ODIN merges these with the help of deformable 2D attention, like to Mask2Former (the three lowest resolution scales: 1/32, 1/16, 1/8).

When we have 3D input, we apply an additional 3D fusion layer at each scale after the deformable attention, to restore the 3D consistency.

Finally, ODIN uses a simple upsampling layer on the 1/8 resolution feature map to bring it to 1/4 resolution and add with a skip connection to the 1/4 feature map from the backbone.



# ODIN architecture – deeper look

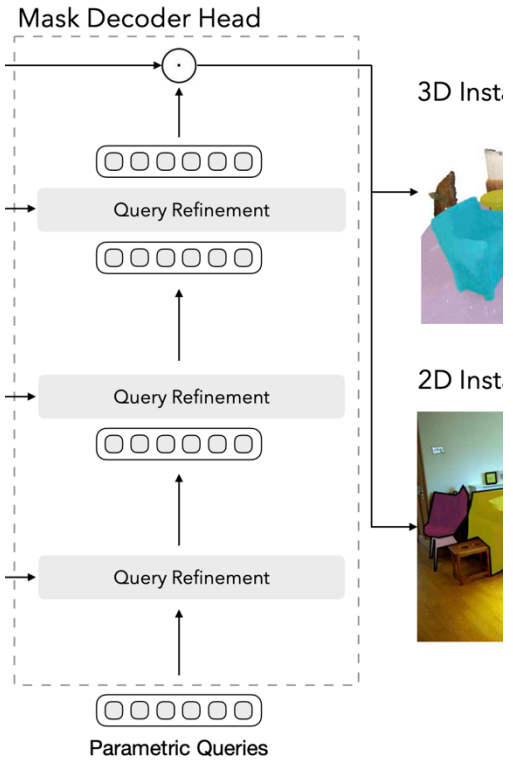
## Shared 2D-3D segmentation mask decoder

ODIN segmentation decoder is a Transformer which takes as input upsampled 2D or 3D feature maps and outputs corresponding 2D or 3D segmentation masks and their semantic classes.

Specifically, instantiate a set of N learnable object queries responsible for decoding individual instances.

These queries are iteratively refined by a Query Refinement block, which consists of cross-attention to the upsampled features, followed by a self-attention between the queries.

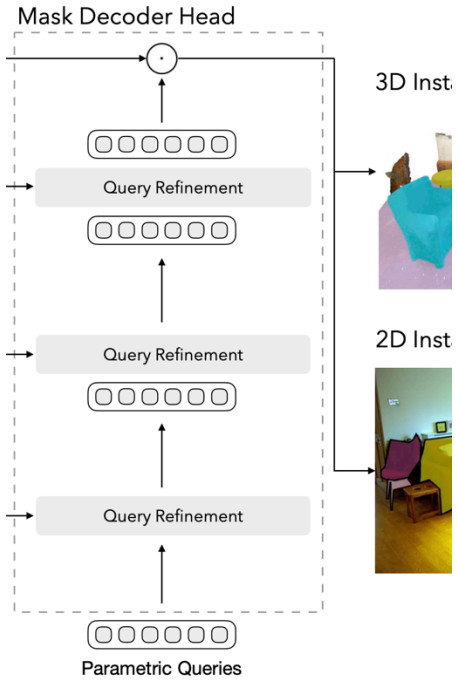
Except for the positional embeddings, all attention and query weights are shared between 2D and 3D.



# ODIN architecture – deeper look

Similar to BUTD-DETR and GLIP

**Shared 2D-3D segmentation mask decoder: the open vocabulary class decoder**



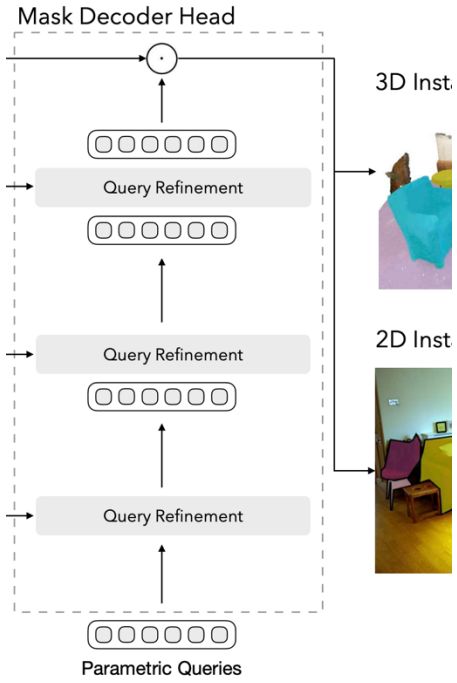

This modification is essential for joint training on multiple datasets.

ODIN supplies the model with a detection prompt formed by concatenating object categories into a sentence (e.g., “Chair. Table. Sofa.”) and encode it using RoBERTa.

In the query-refinement block, queries additionally attend to these text tokens before attending to the upsampled feature maps.

# ODIN architecture – deeper look

**Shared 2D-3D segmentation mask decoder: the open vocabulary class decoder**

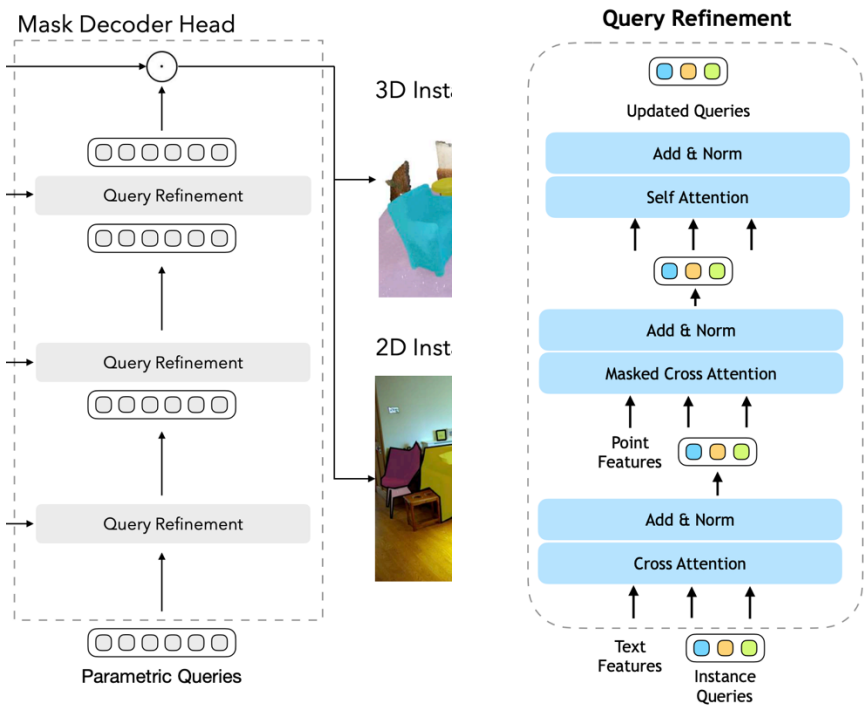



For semantic class prediction, ODIN first performs a dot-product operation between queries and language tokens, generating one logit per token in the detection prompt.

This can handle multi-word noun phrases such as “shower curtain”, where we average the logits corresponding to “shower” and “curtain”.

# ODIN architecture – deeper look

**Shared 2D-3D segmentation mask decoder: the query refinement block**




Queries first attend to the text tokens, then to the visual tokens and finally undergo self-attention.

The text features are optional and are only used in the open-vocabulary decoder setup.

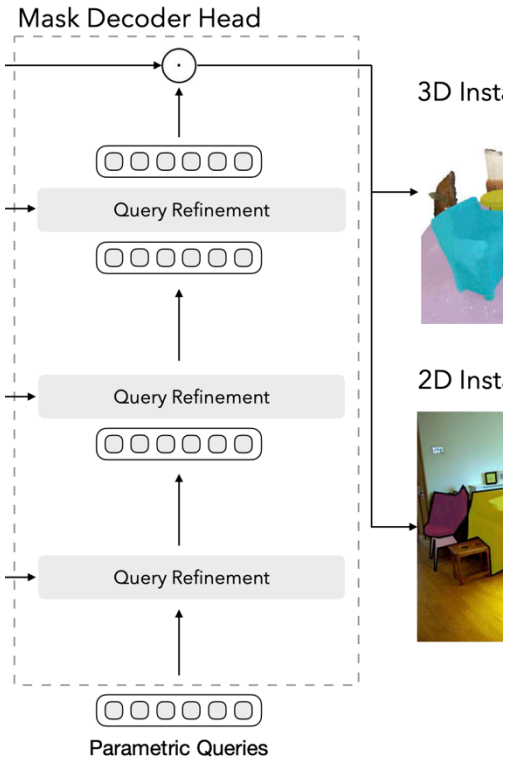
# ODIN architecture – deeper look

## Shared 2D-3D segmentation mask decoder

The refined queries are used to predict instance masks and semantic classes:

- For mask prediction, the queries do a token-wise dot product with the highest-resolution upsampled features.
- For semantic class prediction, ODIN uses an MLP over the queries, mapping them to class logits.

ODIN uses Hungarian matching for matching queries to ground truth instances and supervision losses.

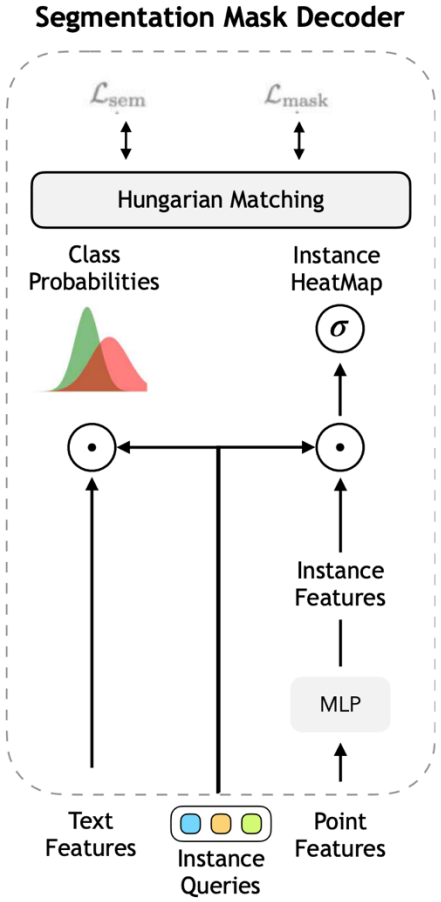


# ODIN architecture – deeper look

**Shared 2D-3D segmentation mask decoder**

This is the is the segmentation mask decoder head where the queries simply perform a dot-product with visual tokens to decode the segmentation heatmap, which can be thresholded to obtain the segmentation mask.

Mask2Former's strategy

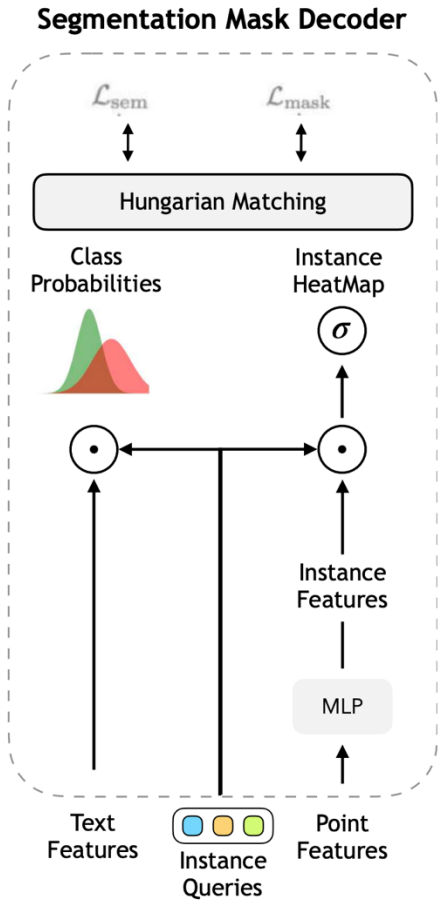


# ODIN architecture – deeper look

**Shared 2D-3D segmentation mask decoder**

In the Open-Vocabulary decoding setup, the queries also perform a dot-product with text tokens to decode a distribution over individual words.

In a closed vocabulary decoding setup, queries simply pass through an MLP to predict a distribution over classes.



# Experiments

# Experiments

**Setup**:

- Initialize the model with pre-trained weights from Mask2Former trained on COCO.
- Subsequently, train all parameters end-to-end, including both pre-trained and new parameters from 3D fusion layers.

# Experiments

**Datasets**:

- ScanNet
- ScanNet200

**Evaluation metrics**:

- mean Average Precision (mAP) for instance segmentation
- mean Intersection over Union (mIoU) for semantic segmentation.

# Experiments

(a) **ScanNet Instance Segmentation Task.**

| | Model | mAP | mAP50 | mAP25 |
|---|---|---|---|---|
| Sensor RGBD Point Cloud | Mask3D[§] [44] | 43.9 | 60.0 | 69.9 |
| | ODIN-ResNet50 (Ours) | 47.8 | 69.8 | **83.6** |
| | ODIN-Swin-B (Ours) | **50.0** | **71.0** | **83.6** |
| Mesh Sampled Point Cloud | SoftGroup [49] | 46.0 | 67.6 | 78.9 |
| | PBNet [58] | 54.3 | 70.5 | 78.9 |
| | Mask3D [44] | 55.2 | 73.7 | **83.5** |
| | QueryFormer [34] | 56.5 | 74.2 | 83.3 |
| | MAFT [28] | **58.4** | **75.9** | - |

**Performance drops with sensor point cloud as input**: Mask3D's performance drops from 55.2% mAP with mesh point cloud input to 43.9% mAP with sensor point cloud input, misalignments caused by noise in camera poses, depth variations and post-processing steps.

# Experiments

(a) **ScanNet Instance Segmentation Task.**

| | Model | mAP | mAP50 | mAP25 |
|---|---|---|---|---|
| Sensor RGBD Point Cloud | Mask3D[§] [44] | 43.9 | 60.0 | 69.9 |
| | ODIN-ResNet50 (Ours) | 47.8 | 69.8 | **83.6** |
| | ODIN-Swin-B (Ours) | **50.0** | **71.0** | **83.6** |
| Mesh Sampled Point Cloud | SoftGroup [49] | 46.0 | 67.6 | 78.9 |
| | PBNet [58] | 54.3 | 70.5 | 78.9 |
| | Mask3D [44] | 55.2 | 73.7 | **83.5** |
| | QueryFormer [34] | 56.5 | 74.2 | 83.3 |
| | MAFT [28] | **58.4** | **75.9** | - |

**ODIN outperforms SOTA 3D methods with sensor point cloud input and underperforms them when baselines use mesh-sampled point clouds**: outperforms SOTA Mask3D model with sensor point cloud input and achieves comparable performance to methods using mesh-sampled point cloud input on the mAP25 metric while far behind on mAP metric, due to misalignments between the 3D mesh and the sensor point cloud.

# Experiments

(b) **ScanNet Semantic Segmentation Task.**

| | Model | mIoU |
|---|---|---|
| Sensor RGBD Point Cloud | MVPNet [20] | 68.3 |
| | BPNet [17] | 69.7 |
| | DeepViewAgg [40] | 71.0 |
| | ODIN-ResNet50 (Ours) | 73.3 |
| | ODIN-Swin-B (Ours) | **77.8** |
| Rendered RGBD Point Cloud | VMVF [26] | **76.4** |
| Mesh Sampled Point Cloud | Point Transformer v2 [51] | 75.4 |
| | Stratified Transformer [27] | 74.3 |
| | OctFormer [50] | 75.7 |
| | Swin3D-L [55] | **76.7** |
| Zero-Shot | OpenScene [37] | 54.2 |

**ODIN sets a new SOTA in semantic segmentation on ScanNet**: outperforming all methods on all setups including the models trained on the sensor, rendered and mesh sampled point clouds.

# Experiments

(c) **ScanNet200 Instance Segmentation Task.**

| | Model | mAP | mAP50 | mAP25 |
|---|---|---|---|---|
| Sensor RGBD Point Cloud | Mask3D [44] § | 15.5 | 21.4 | 24.3 |
| | ODIN-ResNet50 (Ours) | 25.6 | 36.9 | 43.8 |
| | ODIN-Swin-B (Ours) | **31.5** | **45.3** | **53.1** |
| Mesh Sampled Point Cloud | Mask3D [44] | 27.4 | 37.0 | 42.3 |
| | QueryFormer [34] | 28.1 | 37.1 | **43.4** |
| | MAFT [28] | **29.2** | **38.2** | 43.3 |
| Zero-Shot | OpenMask3D [47] | 15.4 | 19.9 | 23.1 |

**ODIN sets a new instance segmentation SOTA on the long-tailed ScanNet200 dataset:** outperforming SOTA 3D models on all setups including the models trained on mesh-sampled point cloud.

This highlights the contribution of 2D features, particularly in detecting a long tail of class distribution where limited 3D data is available.

# Experiments

(d) **ScanNet200 Semantic Segmentation Task.**

| | Model | mIoU |
|---|---|---|
| Sensor RGBD Point Cloud | ODIN-ResNet50 (Ours) | 35.8 |
| | ODIN-Swin-B (Ours) | **40.5** |
| Mesh Sampled Point Cloud | LGround [41] | 28.9 |
| | CeCo [60] | 32.0 |
| | Octformer [50] | **32.6** |

**ODIN sets a new semantic segmentation SOTA on ScanNet200** : outperforming SOTA semantic segmentation models that use mesh point clouds.

# Experiments

**ODIN outperforms SOTA 3D models on Matterport3D Instance Segmentation Benchmark across all settings ODIN sets a new state-of-the-art on Matterport3D Semantic Segmentation Benchmark**

(a) **Comparison on Matterport3D for Instance Segmentation Task.**

| | | 21 | | 160 | |
|---|---|---|---|---|---|
| Input | Model | mAP | mAP25 | mAP | mAP25 |
| Sensor RGBD Point Cloud | Mask3D [44] | 7.2 | 16.8 | 2.5 | 10.9 |
| | ODIN-ResNet50 (Ours) | 22.5 | 56.4 | 11.5 | 27.6 |
| | ODIN-Swin-B (Ours) | **24.7** | **63.8** | **14.5** | **36.8** |
| Mesh Sampled Point Cloud | Mask3D [44] | **22.9** | **55.9** | **11.3** | **23.9** |

# Experiments

**ODIN achieves superior performance in both the 21 and 160 class settings**

(b) **Comparison on Matterport3D for Semantic Segmentation Task.**

| | | 21 | | 160 | |
|---|---|---|---|---|---|
| Input | Model | mIoU | mAcc | mIoU | mAcc |
| Sensor RGBD Point Cloud | ODIN-ResNet50 (Ours) | 54.5 | 65.8 | 22.4 | 28.5 |
| | ODIN-Swin-B (Ours) | **57.3** | **69.4** | **28.6** | **38.2** |
| Mesh Sampled Point Cloud | TextureNet [18] | - | 63.0 | - | - |
| | DCM-Net [43] | - | **67.2** | - | - |
| | MinkowskiNet [5] | **54.2** | 64.6 | - | **18.4** |
| Zero-Shot | OpenScene [37] | **42.6** | **59.2** | - | **23.1** |

# Experiments

In the setup where baseline Mask3D start from ScanNet pre-trained checkpoint, ODIN outperforms them in the RGBD point cloud setup, but obtains lower performance compared to mesh sampled point cloud methods and when compared on the setup where all models train from scratch.

(c) **Comparison on S3DIS Area5 for Instance Segmentation Task.** († = uses additional data)

| | Model | mAP | mAP50 | mAP25 |
|---|---|---|---|---|
| RGBD Point Cloud | Mask3D [44] | 40.7 | 54.6 | 64.2 |
| | Mask3D [44] † | 41.3 | 55.9 | 66.1 |
| | ODIN-ResNet50 (Ours) | 36.3 | 48.0 | 61.2 |
| | ODIN-ResNet50 † (Ours) | **44.7** | **57.7** | 67.5 |
| | ODIN-Swin-B † (Ours) | 43.0 | 56.4 | **70.0** |
| Mesh Sampled Point Cloud | SoftGroup [49] † | 51.6 | 66.1 | - |
| | Mask3D [44] | 56.6 | 68.4 | 75.2 |
| | Mask3D [44] † | **57.8** | **71.9** | **77.2** |
| | QueryFormer [34] | 57.7 | 69.9 | - |
| | MAFT [28] | - | 69.1 | 75.7 |

# Experiments

On S3DIS Semantic Segmentation Benchmark, ODIN trained with ScanNet weight initialization outperforms all RGBD point cloud based methods, while achieving competitive performance on mesh sampled point cloud.

When trained from scratch, it is much worse than other baselines. Given the limited dataset size of S3DIS with only 200 training scenes, we observe severe overfitting.

(d) **Comparison on S3DIS for Semantic Segmentation Task.** († = uses additional data)

| Input | Model | mIoU |
|---|---|---|
| RGBD Point Cloud | MVPNet [20] | 62.4 |
| | VMVF [26] | 65.4 |
| | DeepViewAgg [40] | 67.2 |
| | ODIN-ResNet50 (Ours) | 59.7 |
| | ODIN-ResNet50 † (Ours) | 66.8 |
| | ODIN-Swin-B † (Ours) | **68.6** |
| Mesh Sampled Point Cloud | Point Transformer v2 [51] | 71.6 |
| | Stratified Transformer [27] | 72.0 |
| | Swin3D-L [55] † | **74.5** |

# Experiments

## Evaluation on multiview RGB-D in simulation

Table 2. **AI2THOR Semantic and Instance Segmentation.**

| Model | mAP | mAP50 | mAP25 | mIoU |
|---|---|---|---|---|
| Mask3D [44] | 60.6 | 70.8 | 76.6 | - |
| ODIN-ResNet50 (Ours) | 63.8 | **73.8** | **80.2** | **71.5** |
| ODIN-Swin-B (Ours) | **64.3** | 73.7 | 78.6 | 71.4 |

ODIN outperforms Mask3D by 3.7% mAP, showing strong performance in a directly comparable RGB-D setup.

# Experiments

## Embodied Instruction Following

Table 3. **Embodied Instruction Following.** SR = success rate. GC = goal condition success rate.

| | TEACh | | | | ALFRED | | | |
|---|---|---|---|---|---|---|---|---|
| | Unseen | | Seen | | Unseen | | Seen | |
| | SR | GC | SR | GC | SR | GC | SR | GC |
| FILM [35] | - | - | - | - | 30.7 | 42.9 | 26.6 | 38.2 |
| HELPER [42] | 15.8 | 14.5 | 11.6 | 19.4 | 37.4 | 55.0 | 26.8 | 41.2 |
| HELPER + ODIN (OURS) | **18.6** | **18.6** | **13.8** | **26.6** | **47.7** | **61.6** | **33.5** | **47.1** |

HELPER with ODIN as its 3D object detector significantly outperforms HELPER that uses the original 2D detection plus linking perception pipeline.

# Experiments

## Ablations and Variants

Table 4. **Joint Training on Sensor RGB-D point cloud from ScanNet and 2D RGB images from COCO.**

| | ScanNet | | | COCO |
|---|---|---|---|---|
| | mAP | mAP50 | mAP25 | mAP |
| Mask3D [44] | 43.9 | 60.0 | 69.9 | ✗ |
| Mask2Former [4] | ✗ | ✗ | ✗ | **43.7** |
| ODIN (trained in 2D) | ✗ | ✗ | ✗ | 43.6 |
| ODIN (trained in 3D) | 47.8 | 69.8 | **83.6** | ✗ |
| ODIN (trained jointly) | **49.1** | **70.1** | 83.1 | 41.2 |

Joint training yields a 1.3% absolute improvement in 3D, and causes a similar drop in 2D.

Note that we do not jointly train on 2D and 3D datasets in any of our other experiments due to computational constraints.

# Experiments

## Ablations and Variants

(a) **Cross-View Contextualization.**

| Model | mAP | mIoU |
|---|---|---|
| ODIN (Ours) | **47.8** | 73.3 |
| No 3D Fusion | 39.3 | 73.2 |
| No interleaving | 41.7 | **73.6** |

**Cross-View fusion is crucial for instance segmentation but not for semantic segmentation**:

Row-3 shows a 6.1% mAP drop when cross-view 3D fusion happens after all within-view 2D layers instead of interleaving the within-view and cross-view fusion

# Experiments

## Ablations and Variants

(c) **Effect of Freezing Backbone.**

| Model | ResNet50 | | Swin-B | |
|---|---|---|---|---|
| | mAP | mIoU | mAP | mIoU |
| ODIN (Ours) | **47.8** | 73.3 | **50.0** | **77.8** |
| With frozen backbone | 46.7 | **74.3** | 46.2 | 75.9 |

**Stronger 2D backbones helps:**

using Swin-B over ResNet50 leads to significant performance gains, suggesting that ODIN can directly benefit from advancements in 2D computer vision.

# Experiments

## Ablations and Variants

**Supplying 2D features directly to 3D models does not help:**

Concatenating 2D features with XYZ+RGB as input to Mask3D yields 53.8% mAP performance, comparable to 53.3% of the baseline model with only XYZ+RGB as input.

# Experiments

## Qualitative Results

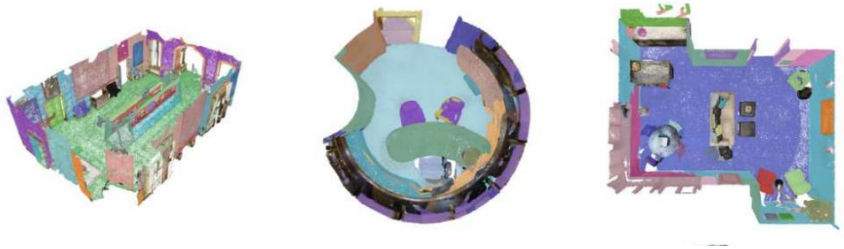
ScanNet

S3DIS

ScanNet200

AI2THOR

Matterport3D



COCO

# Conclusions

# Conclusions

Results:

- SOTA performance in ScanNet200 and AI2THOR instance segmentation benchmarks;
- Outperforms all methods operating on sensor point clouds and achieves competent performance to methods operating over mesh-sampled pointcloud.

# Conclusions

Future work:

- making 3D models more resilient to noise;
- learning by training on diverse 2D and 3D datasets jointly;
- Exploring ways to make 2D and 3D training more synergistic.

# Review

# Review

What is good ?

- Idea (interleaving);
- Results (local);
- Readability;
- Fairness.

# Review

What is NOT good ?

- Not optimazed architecture: inference time, no joint training;
- Lack of technical details;
- Not «real» real time application;
- Use of the «evil» benchmarks;
- «Many inspirations».

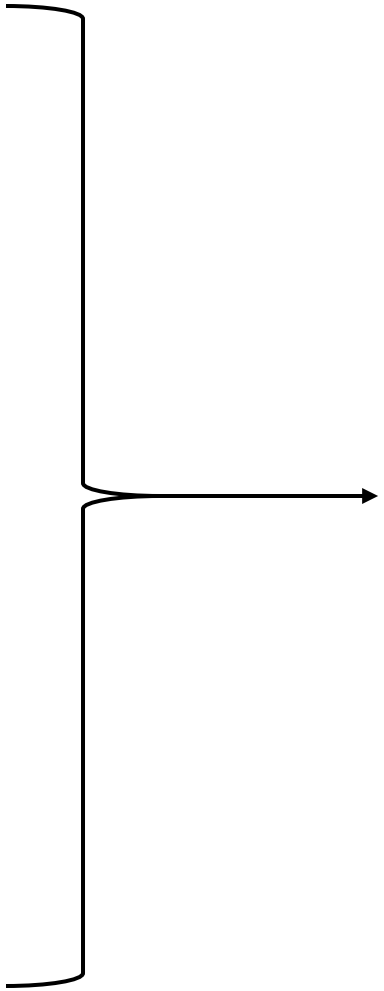
# Review

What is NOT good ?

- Not optimazed architecture: inference time, no joint training;
- Lack of technical details;
- Not «real» real time application;
- Use of the «evil» benchmarks;
- «Many inspirations».

What is good ?

- Idea (interleaving);
- Results (local);
- Readability;
- Fairness.



It is «only» a good proof of concept, because they have just proved their idea, but not in the scenario they had envisaged and using the using the «problematic» benchmarks.

Thank you for the attention