# DenseFusion: 6D Object Pose Estimation by Iterative CVPR-2019

KIST
**송명하**

Korea Institute of Science and Technology

한국과학기술연구원

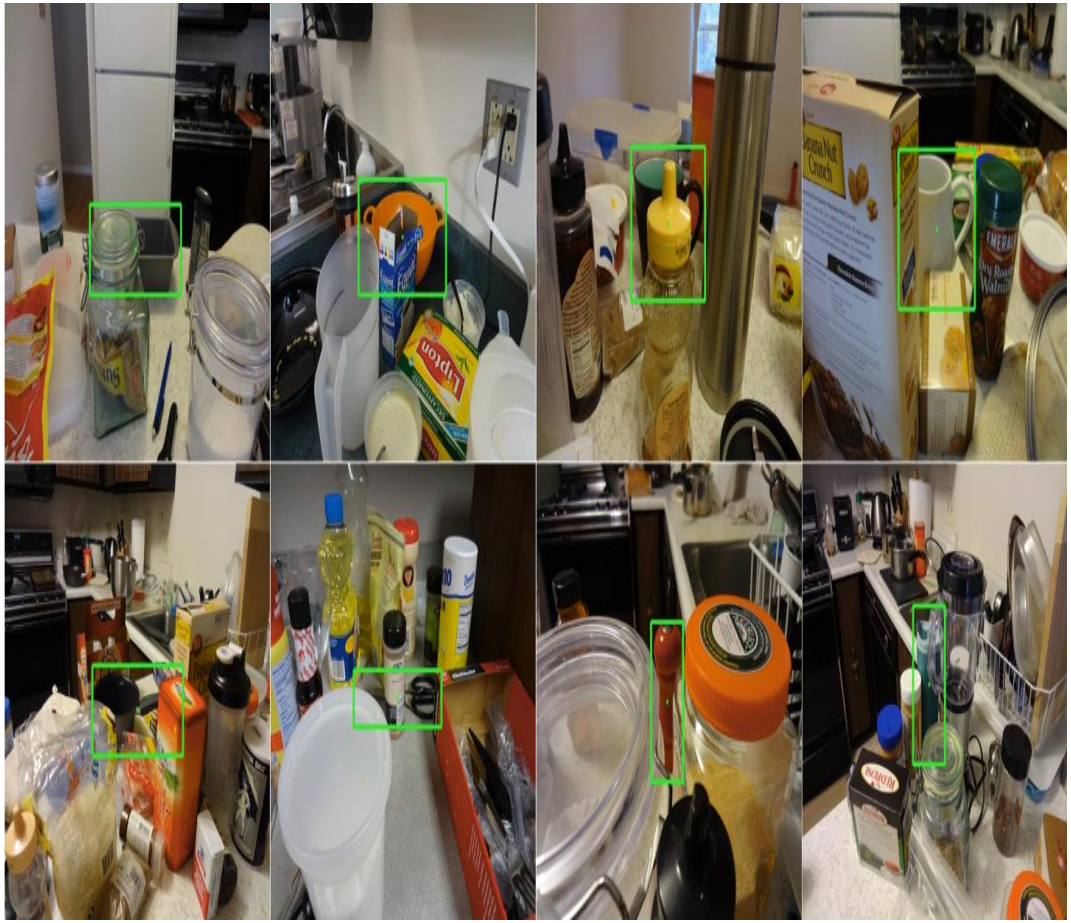
# Content

# 1. Introduction


http://www.autodaily.co.kr/news/articleView.html?idxno=408344

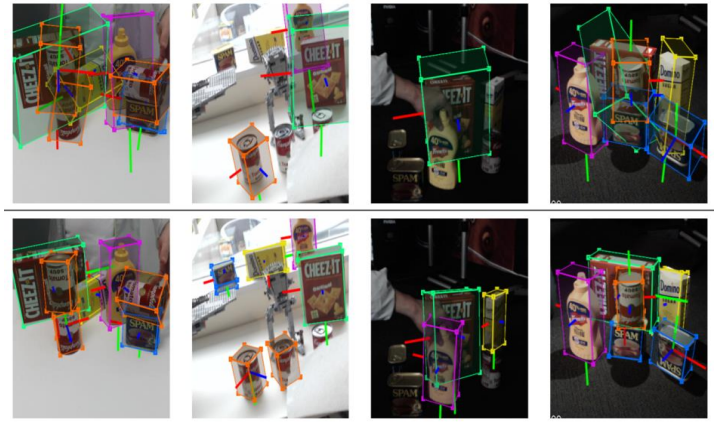
https://www.independent.co.uk/extras/indybest/gadgets-tech/video-games-consoles/augmented-reality-games-ar-android-ios-apple-google-play-tech-a8506831.html

# 1. Introduction



https://www.cs.cmu.edu/~hebert/occarbview.html

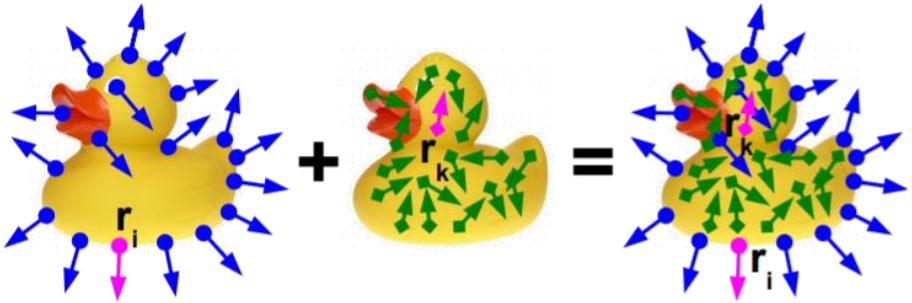
https://www.photoreview.com.au/tips/shooting/how-to-control-image-noise/



Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects

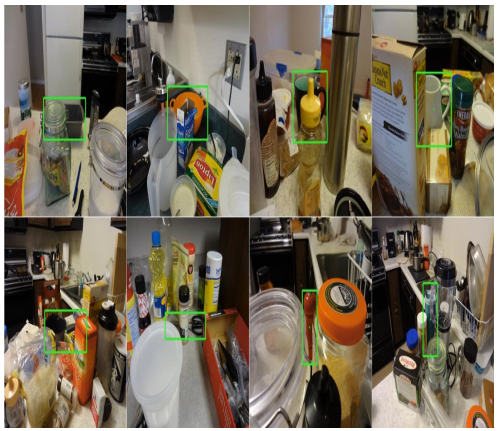
# 1. Introduction

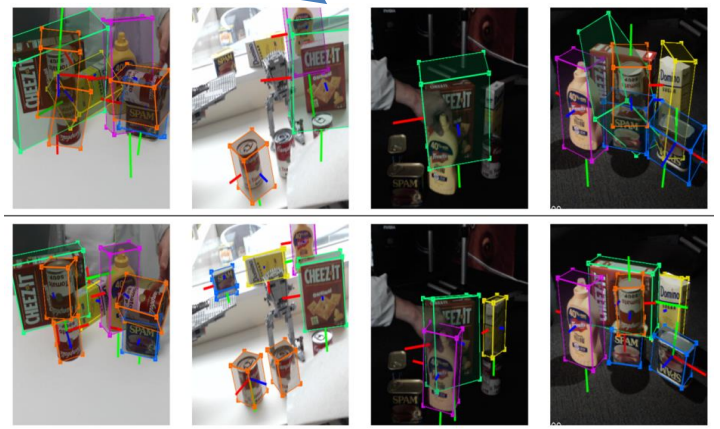
기존 방법들..



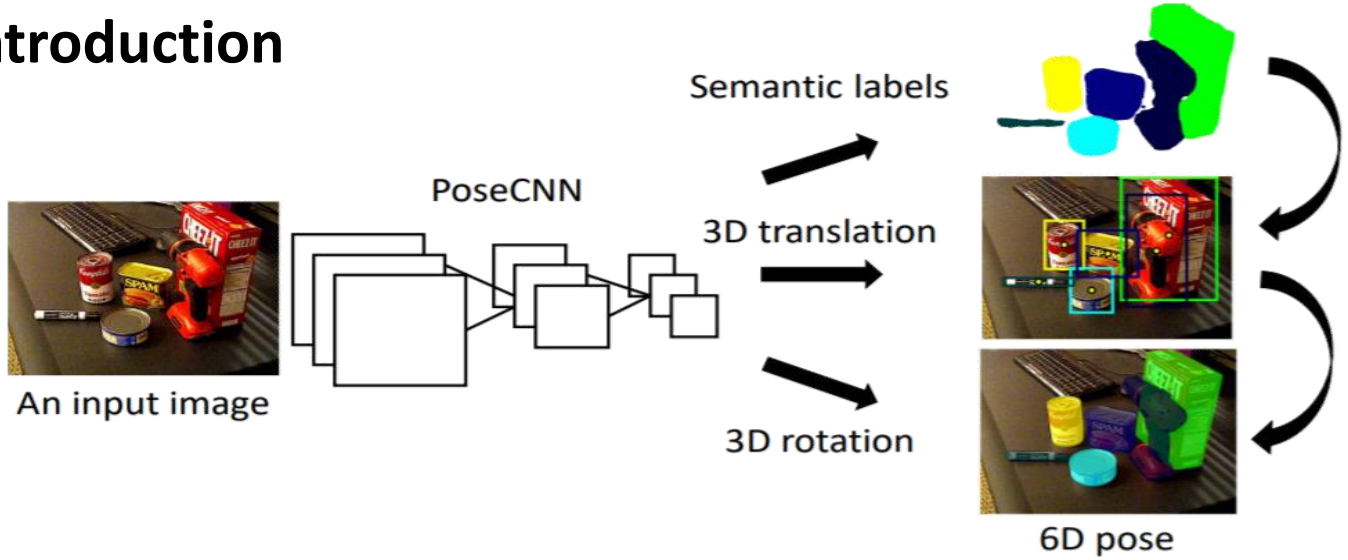

m = gradients  m = surface normals  multiple modalities

Multimodal Templates for Real-Time Detection of Texture-less Objects in Heavily Cluttered Scenes(ICCV2011)

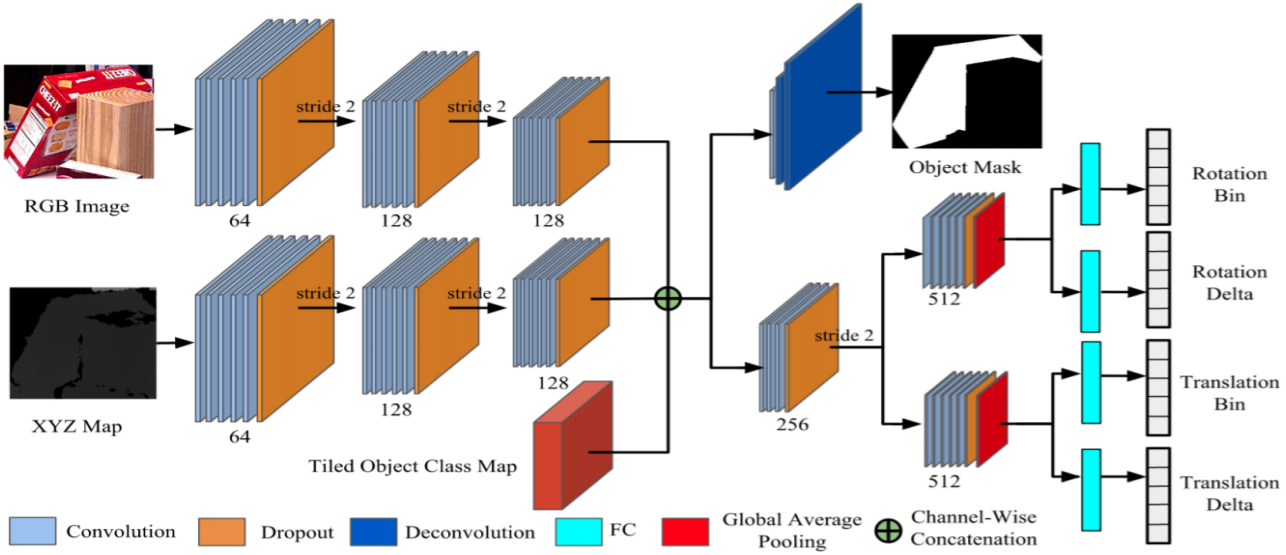




# 1. Introduction

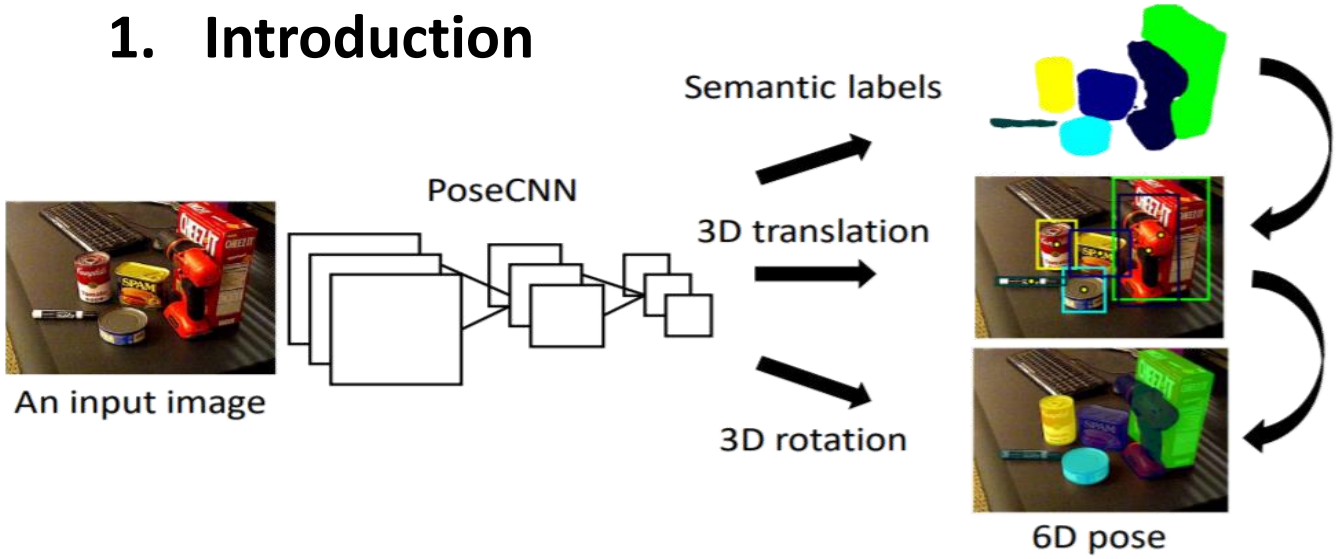



PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes

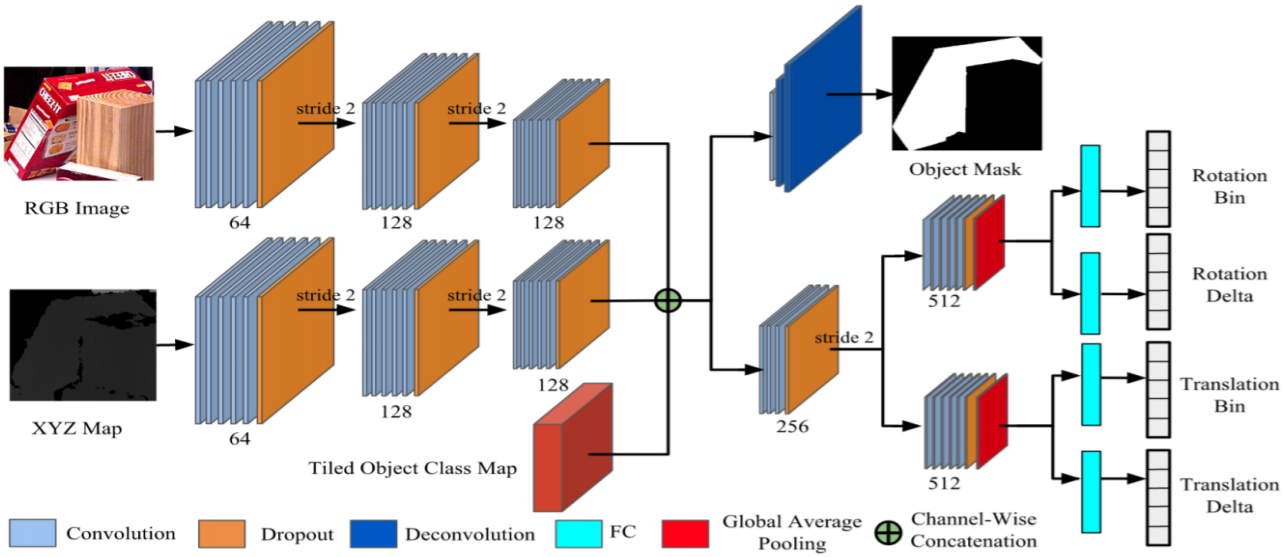



MCN: A Unified Framework for Multi-View Multi-Class Object Pose Estimation

# 1. Introduction



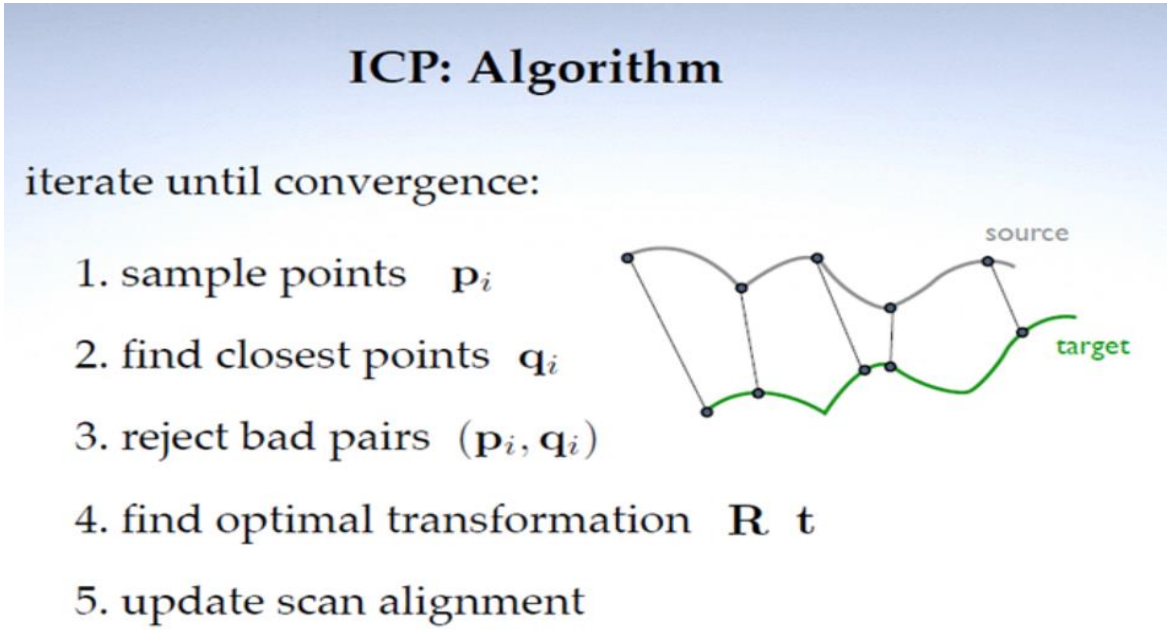

PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes




MCN: A Unified Framework for Multi-View Multi-Class Object Pose Estimation

+ **ICP** **(refinement algorithm)**

# ICP




https://m.blog.naver.com/PostView.nhn?blogId=tlaja&logNo=220666876033&proxyReferer=https%3A%2F%2Fwww.google.com%2F

1.r each point (from the whole set of vertices usually referred to as dense or a selection of pairs of vertices from each model) in the source point cloud, **match the closest point in the reference point cloud (or a selected set).**

2.Estimate the combination of rotation and translation using a root **mean square point to point distance metric** minimization **technique** which will best align each source point to its match found in the previous step. This step may also involve weighting points and rejecting outliers prior to alignment.

3.Transform the source points using the obtained transformation.

4.Iterate (re-associate the points, and so on).

# 1. Introduction

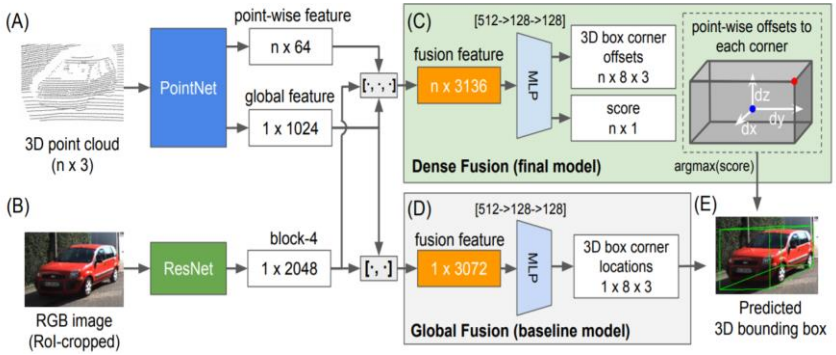
## Contribution

1. We present a principled way to combine color and depth information from the **RGB-D input**. We augment the information of each 3D point with 2D information from an embedding space learned for the task and use this **new color-depth space to estimate the 6D pose.**

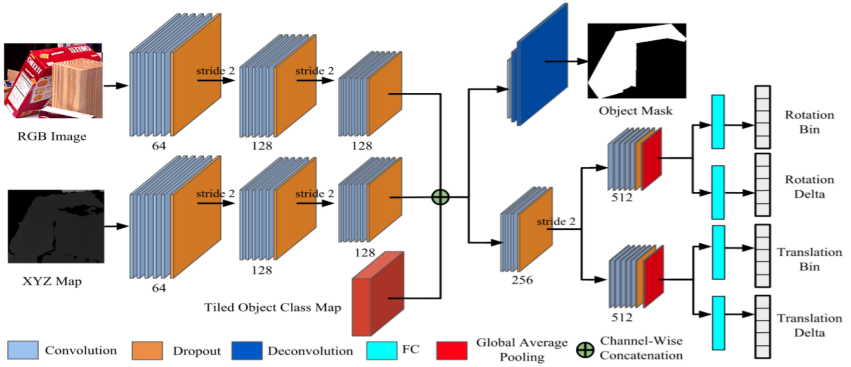
2. **We integrate an iterative refinement procedure** within the neural network architecture, removing the dependency of previous methods of a post-processing ICP step.

# 2. Related Work

## Point Fusion




## MCN




## PoseCNN

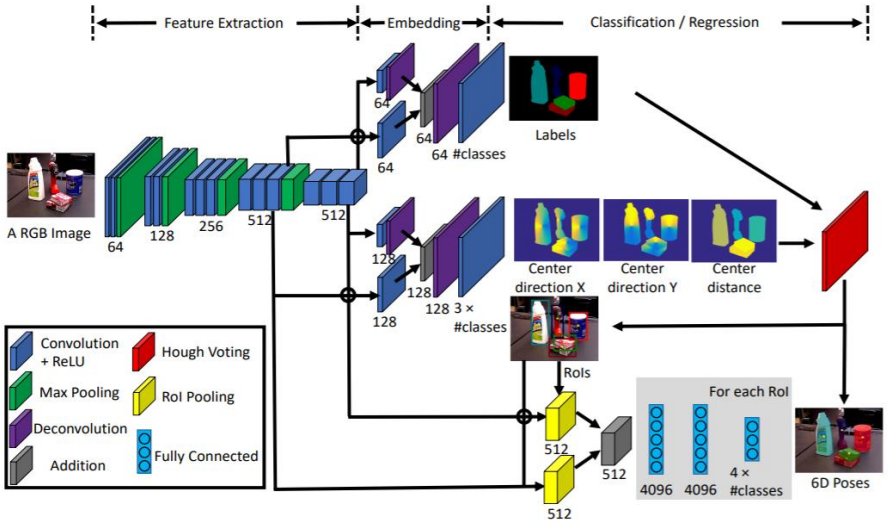
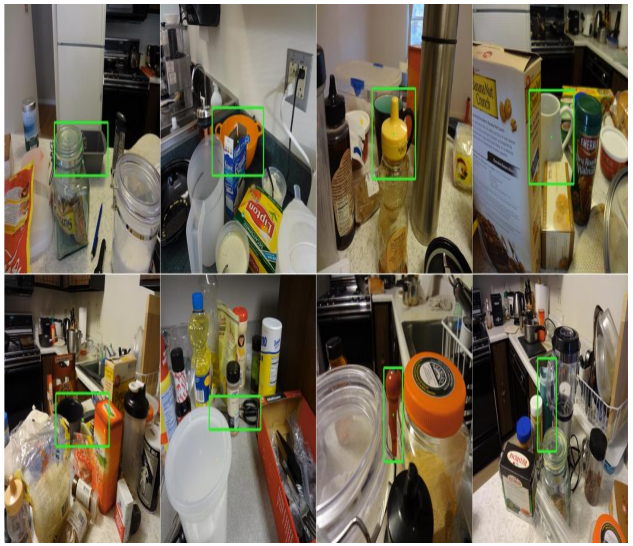


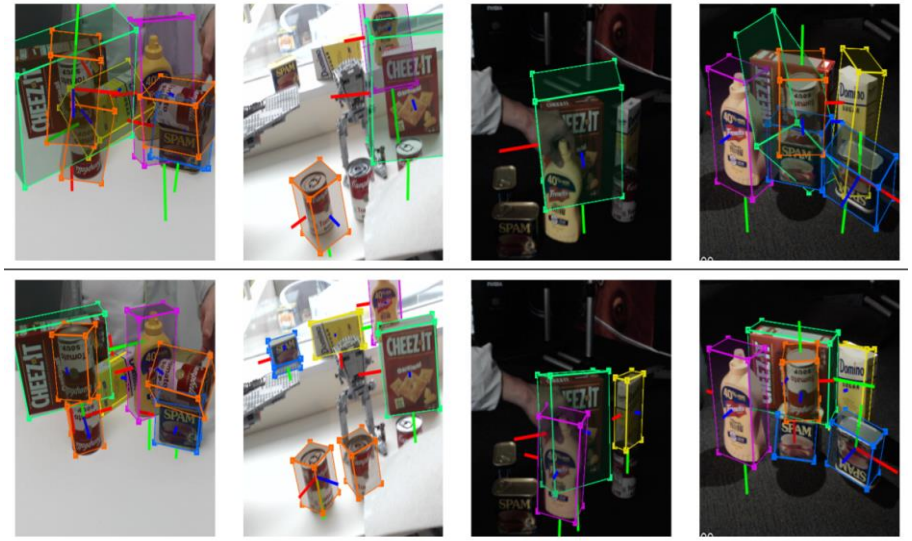

Fig. 2. Architecture of PoseCNN for 6D object pose estimation.

# 3. Model

**우리 목표 : 6D pose = 3D Rotation + 3D translation**

**Challenge : heavy occlusion, lighting condition, real-time inference**

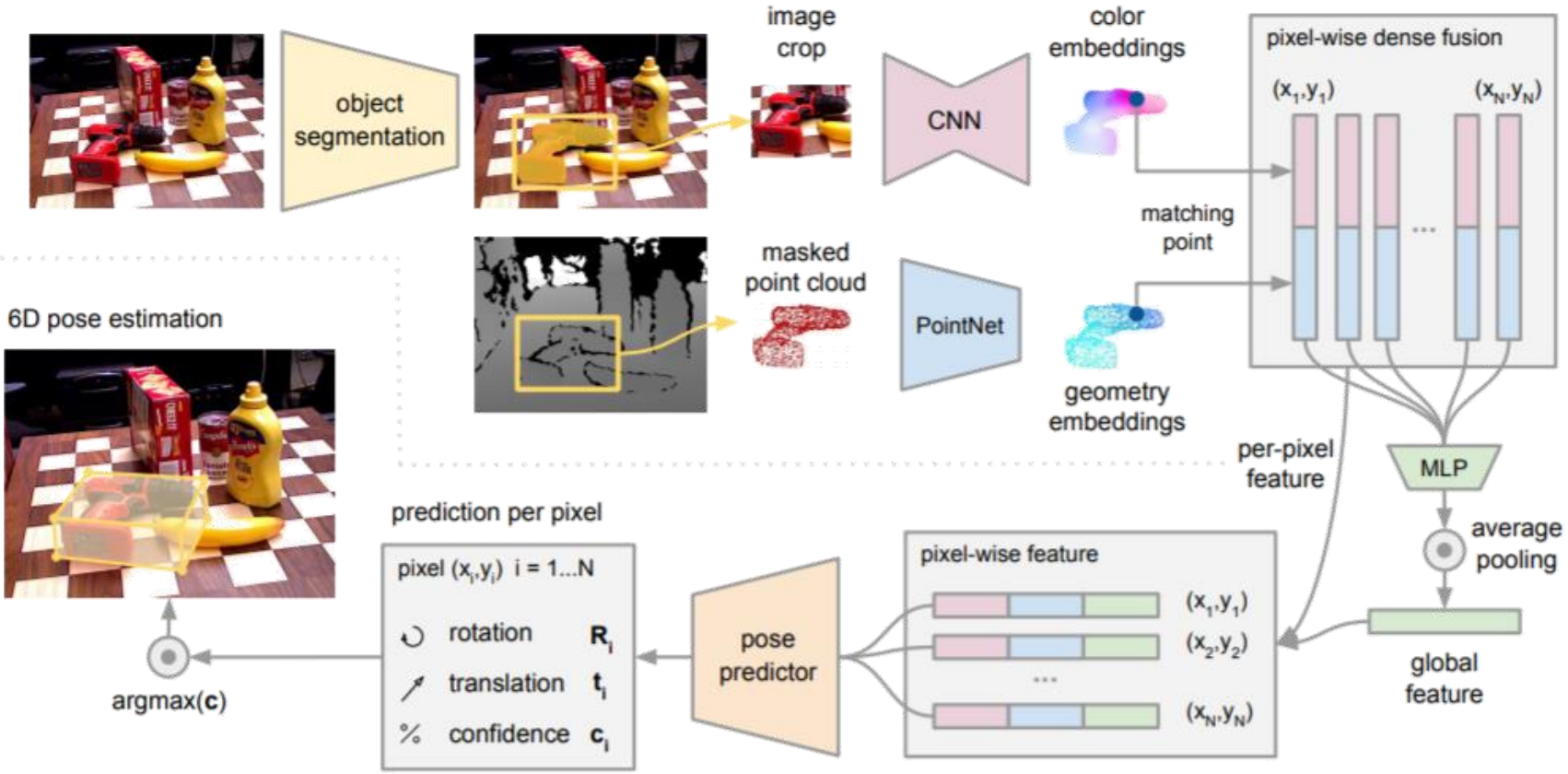




1) RGB information과 Depth information을 적절하게 잘 이용해야한다.
2) Post-hoc 인 refinement 과정의 속도를 빠르게 한다.

# 3. Model

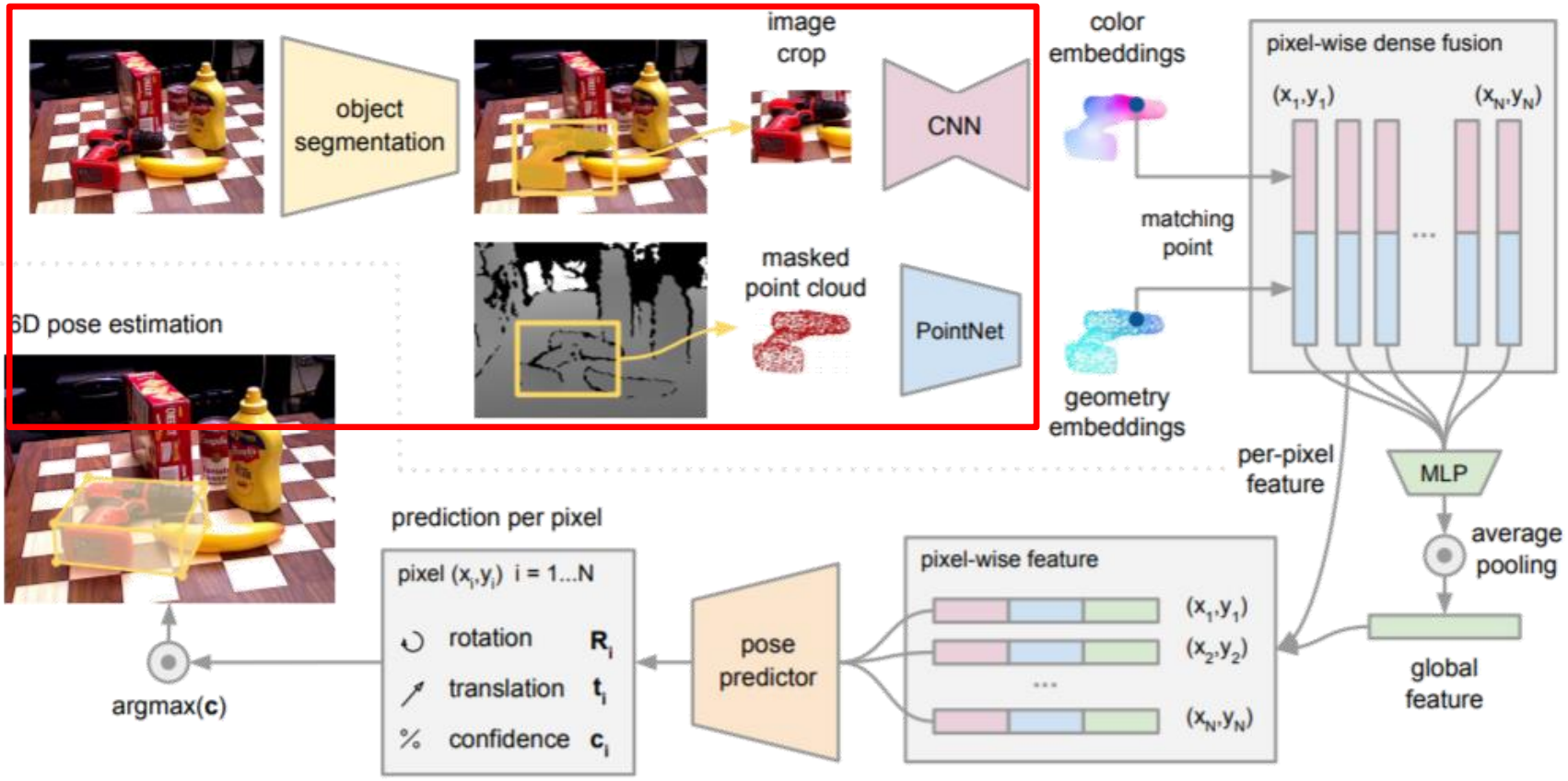
## 3.1 Architecture Overview



# 3. Model

## 3.1 Architecture Overview

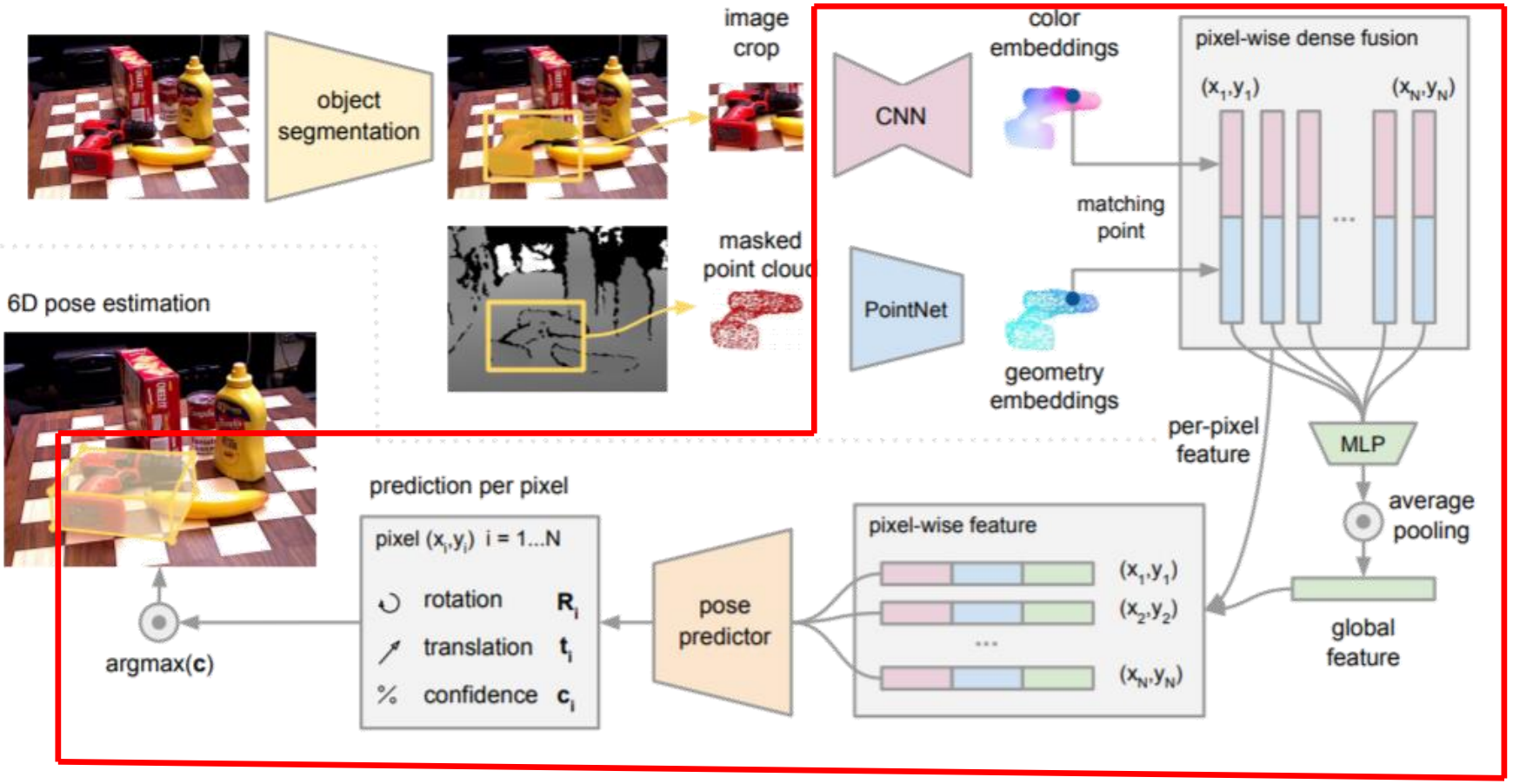
The first stage : take color image as input and performs semantic segmentation for each known object category.



# 3. Model

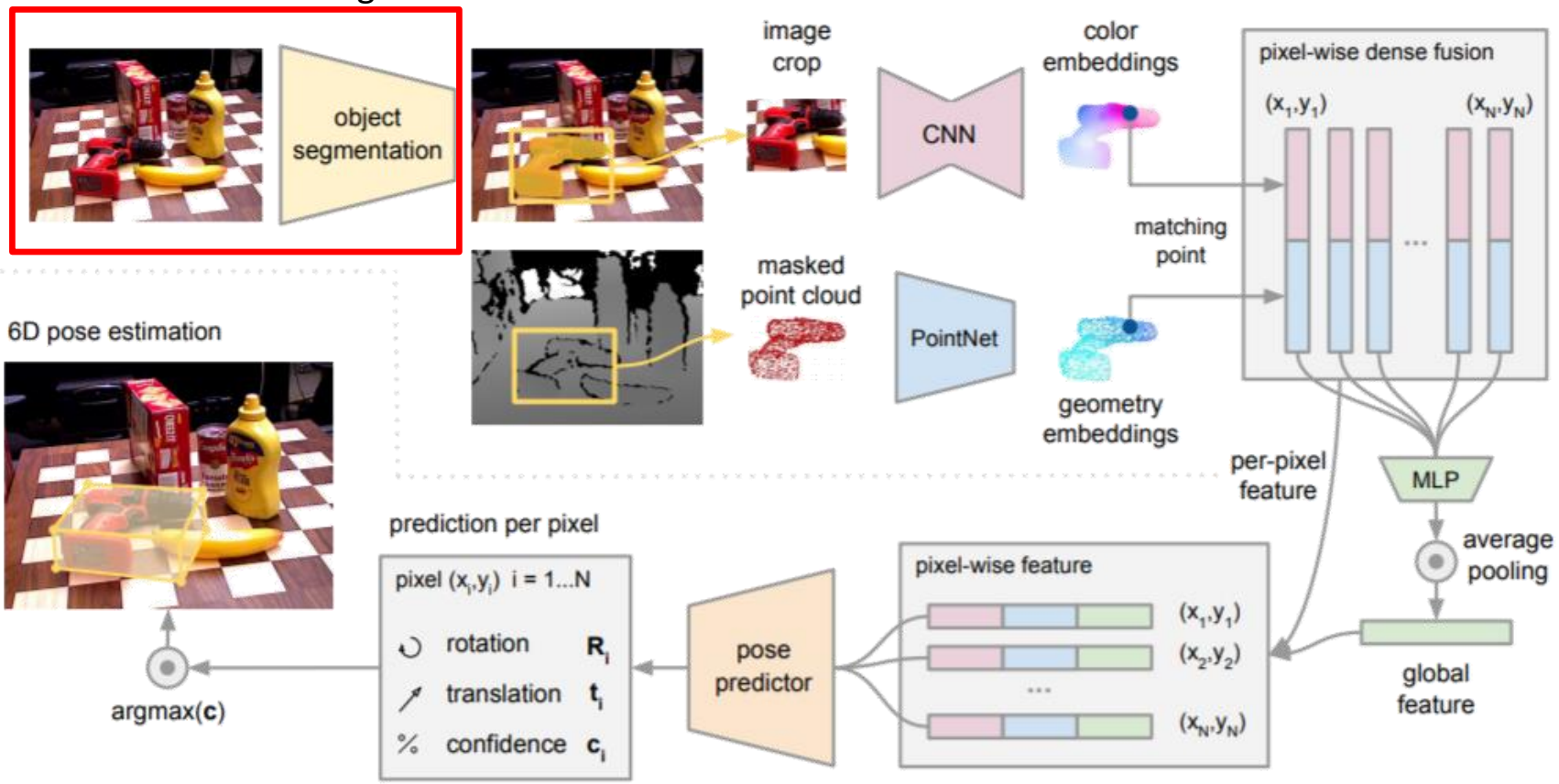
## 3.1 Architecture Overview

The second stage : The second stage processes the results of the segmentation and estimates the object's 6D pose.



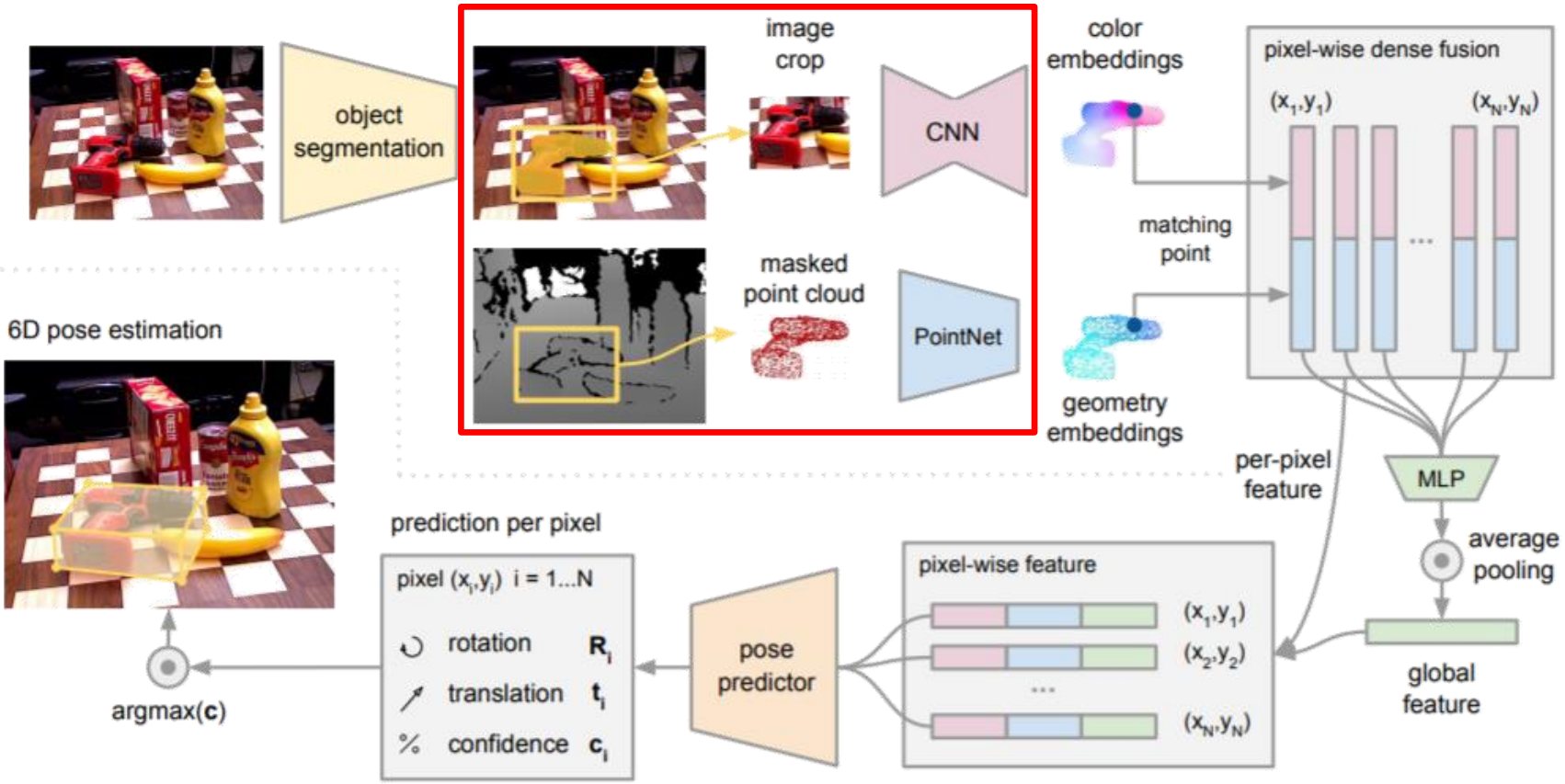
# 3. Model

## 3.2 Semantic Segmentation



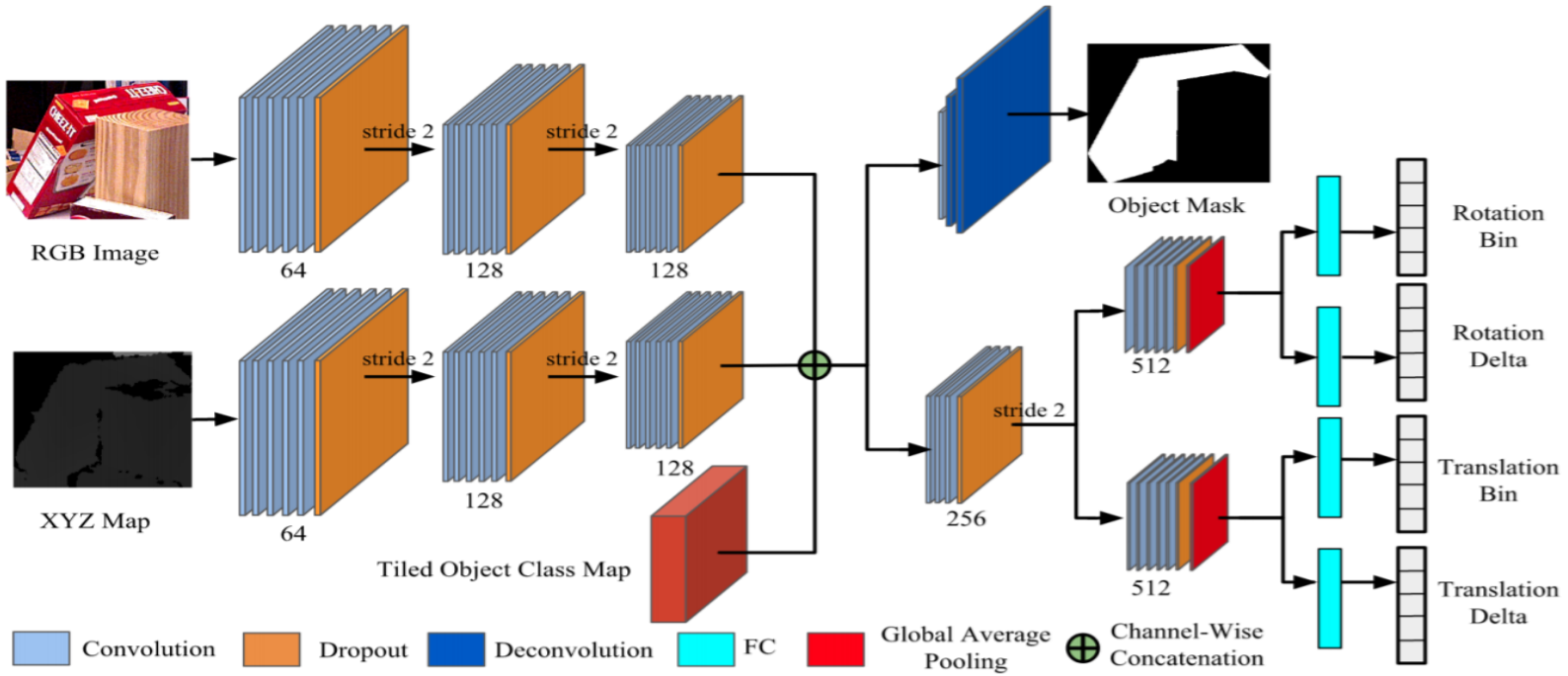
# 3. Model

## 3.3 DenseFeature Extraction



# 3. Model

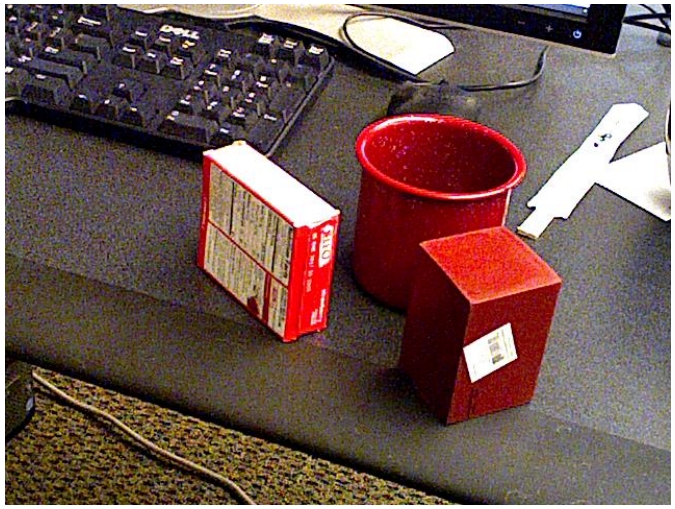
## 3.3 DenseFeature Extraction

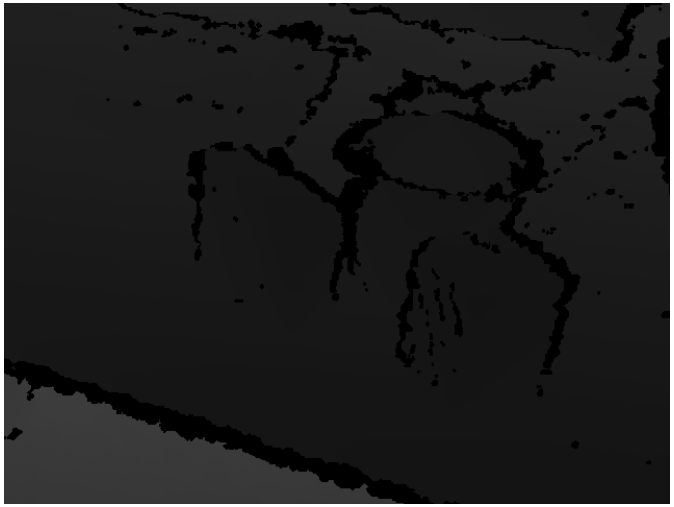



구조적으로 무시하는 것…

# 3. Model

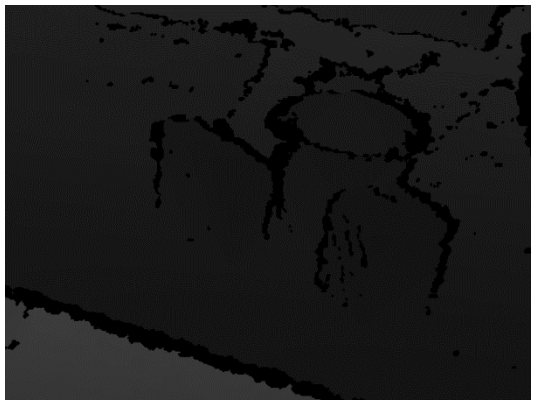
## 3.3 DenseFeature Extraction

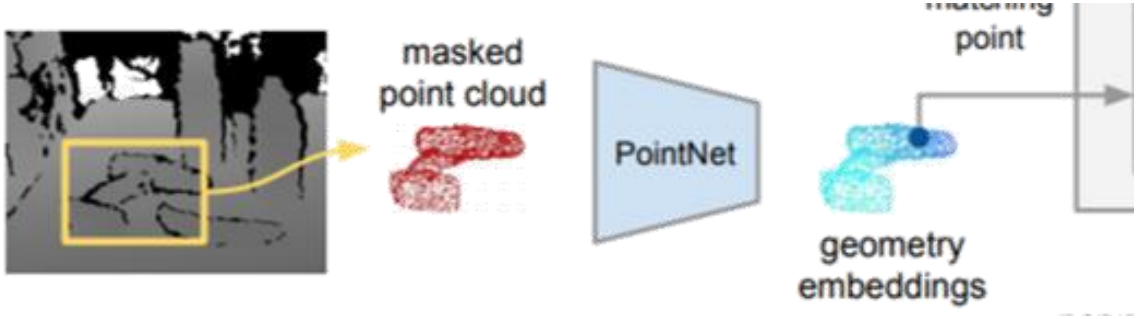




# 3. Model

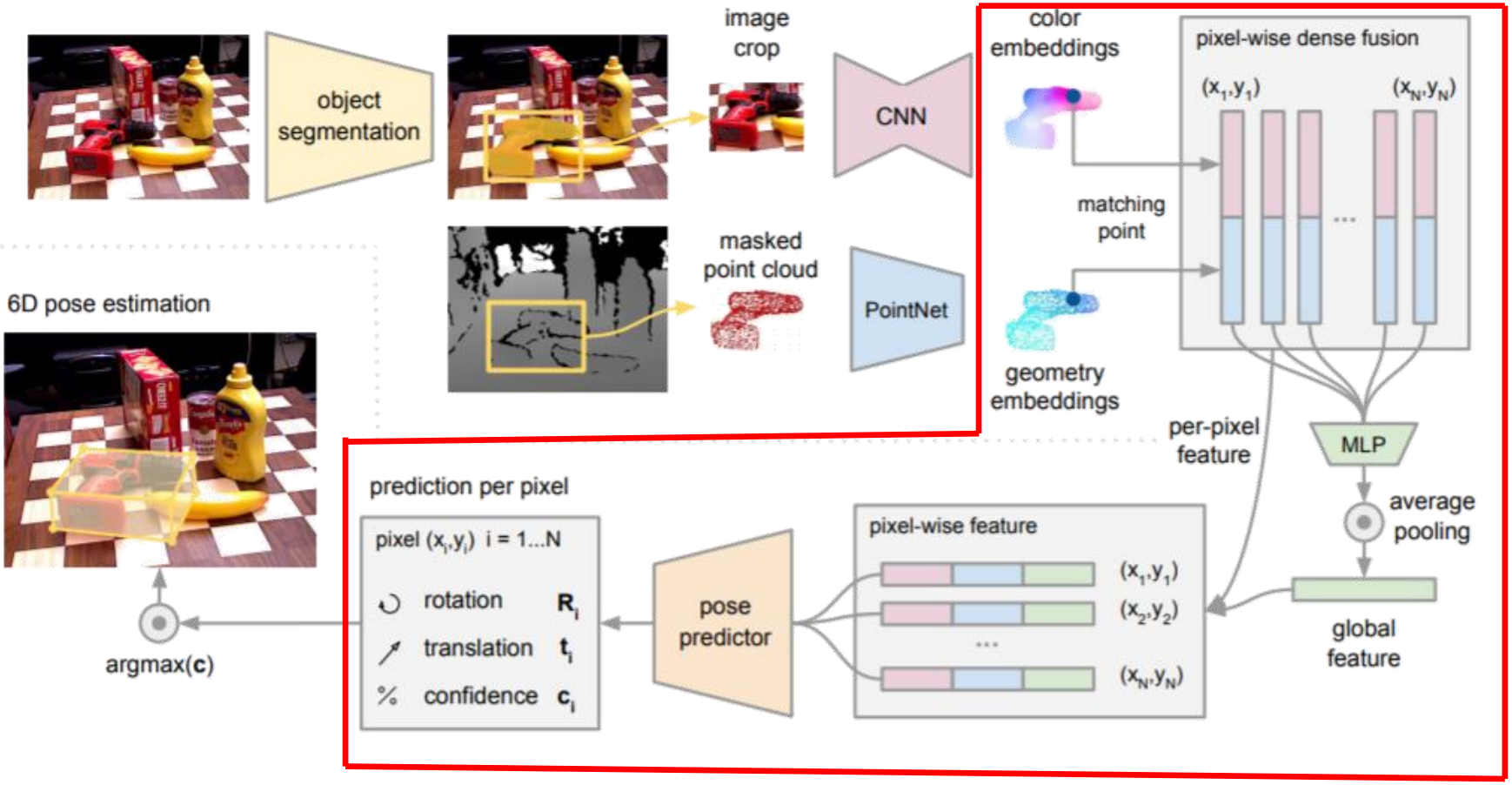
## 3.3 DenseFeature Extraction





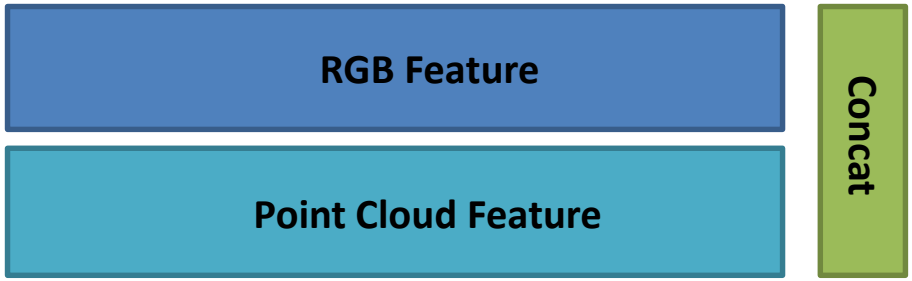
# 3. Model

## 3.4 pixel wise denseFusion



# 3. Model

## 3.4 pixel wise denseFusion

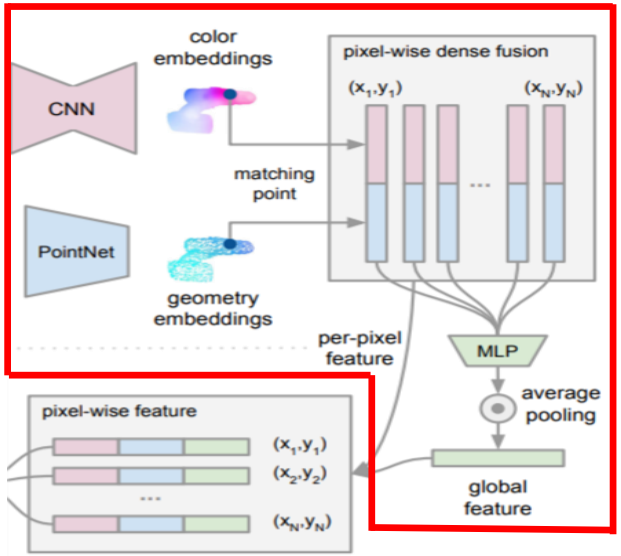


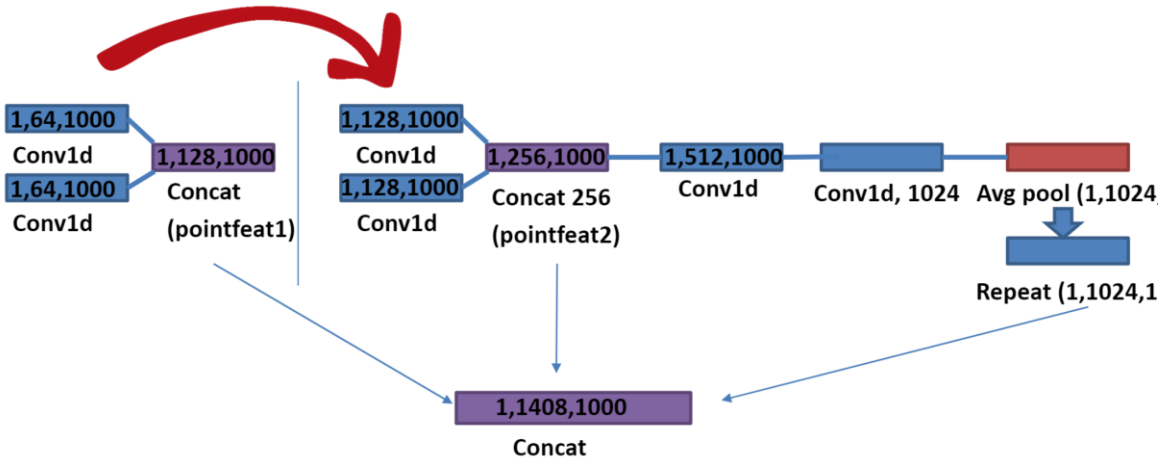

A naïve approach would be to **generate a global feature** from the dense color and depth features from the segmented area.

However, due **to heavy occlusion and segmentation errors**, the set of features from previous step may contain features of points/pixels on objects or parts of the background

# 3. Model

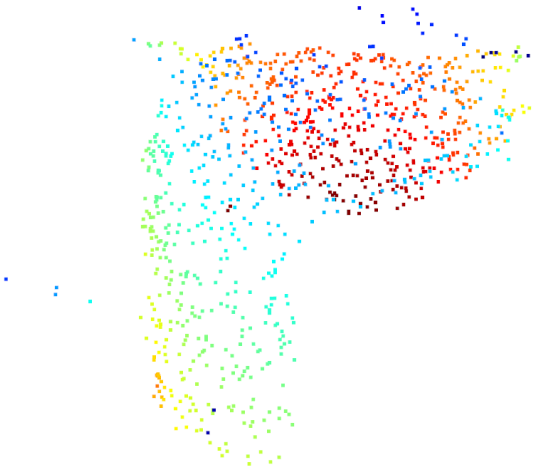
## 3.4 pixel wise denseFusion



# 3. Model

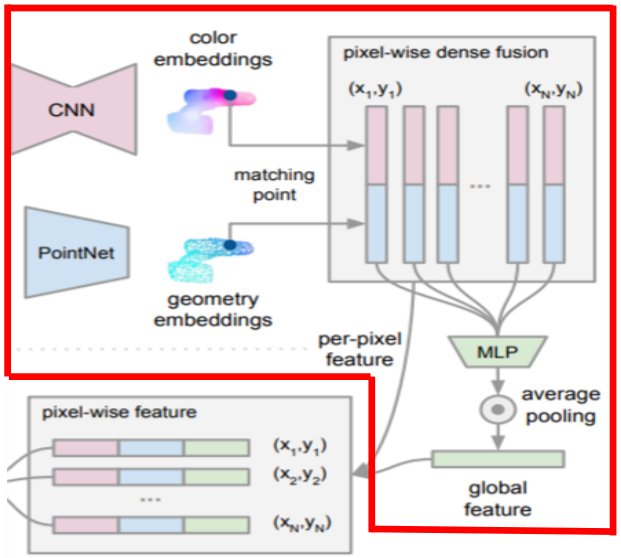
## 3.4 pixel wise denseFusion

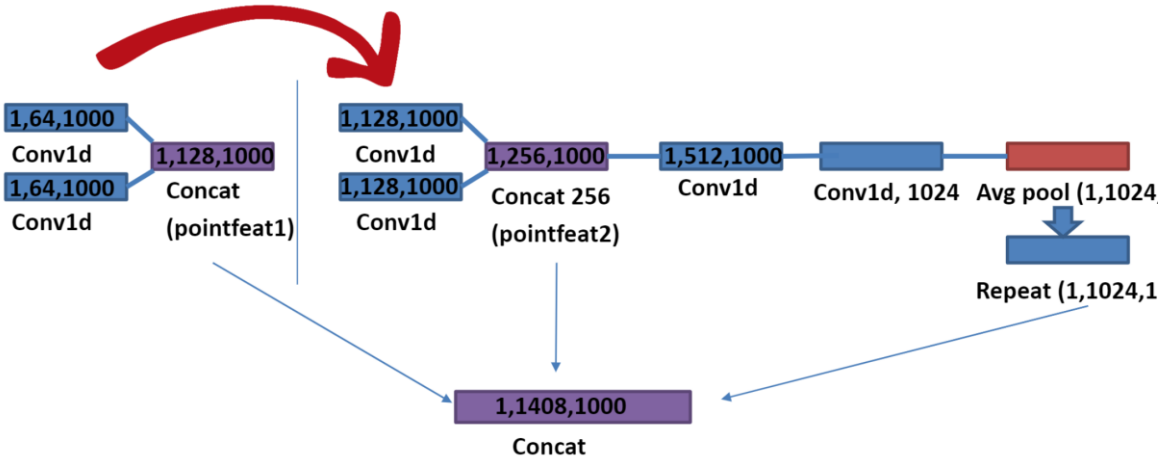


**서로 관련된 Feature 를 뽑기 위해서 random 난수 생성 후 index를 이용하여 Point cloud에서 sampling**

# 3. Model

## 3.4 pixel wise denseFusion



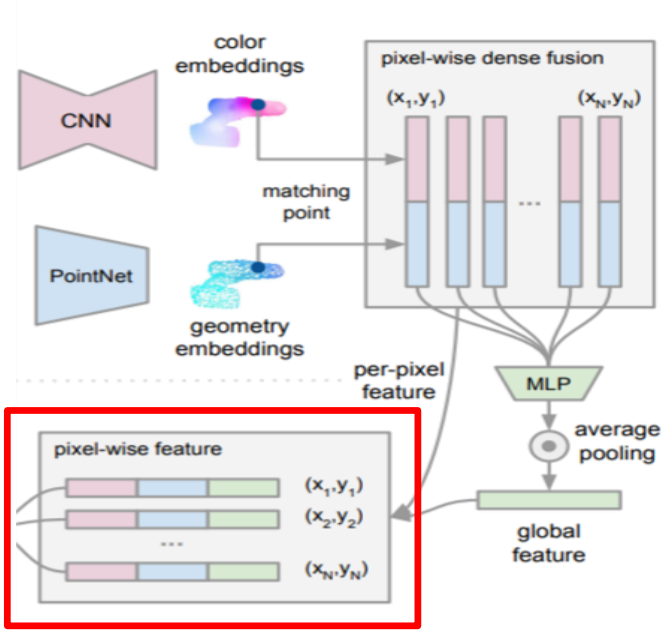



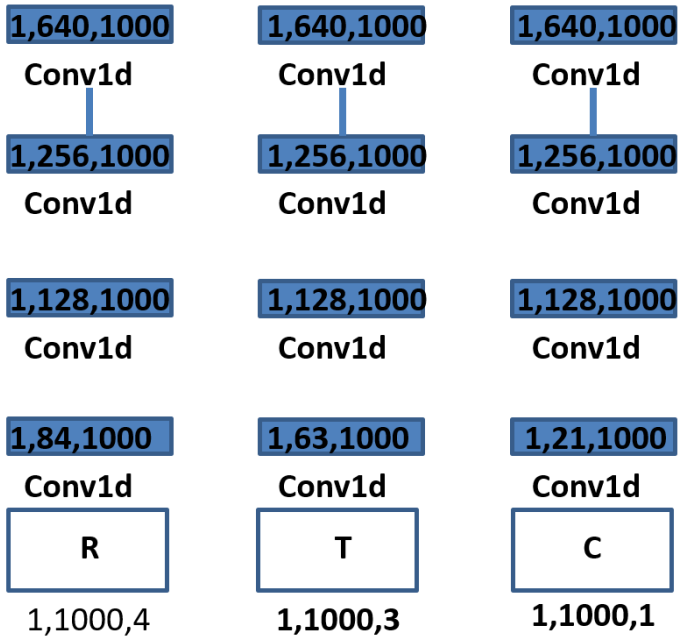

**얻는 효과:**

1. **잠재적으로 물체의 보이는 부분에 기반해서 고르게 되고, occlusion과 segmentation noise를 줄임.**
2. **Local 정보도 풍부한 feature를 얻을 수 있음.**

# 3. Model

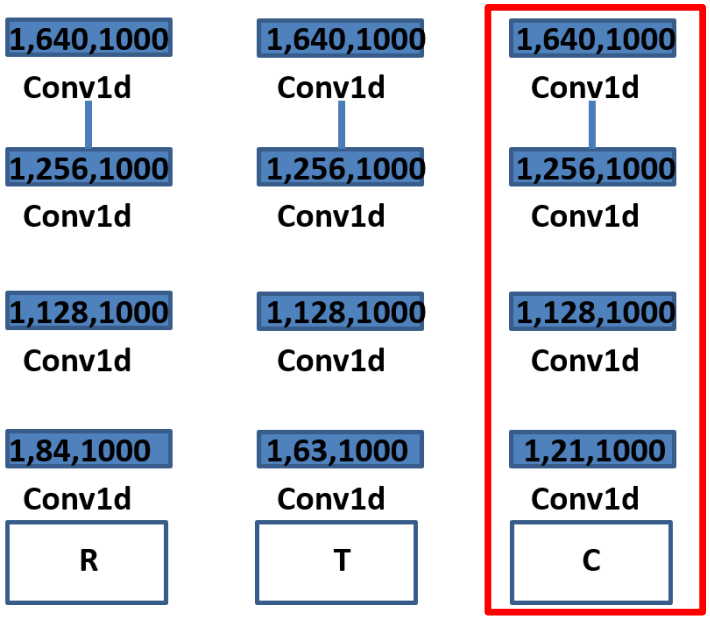
## 3.4 pixel wise denseFusion



# 3. Model

## 3.4 pixel wise denseFusion




we would like to train our pose estimation network to **decide which pose estimation is likely to be the best hypothesis based on the specific context**(inspired by Point Fusion).

**To do so, we modify the network to output a confidence score c, for each prediction in addition to the pose estimation prediction.**

# 3. Model

## 3.5 Loss

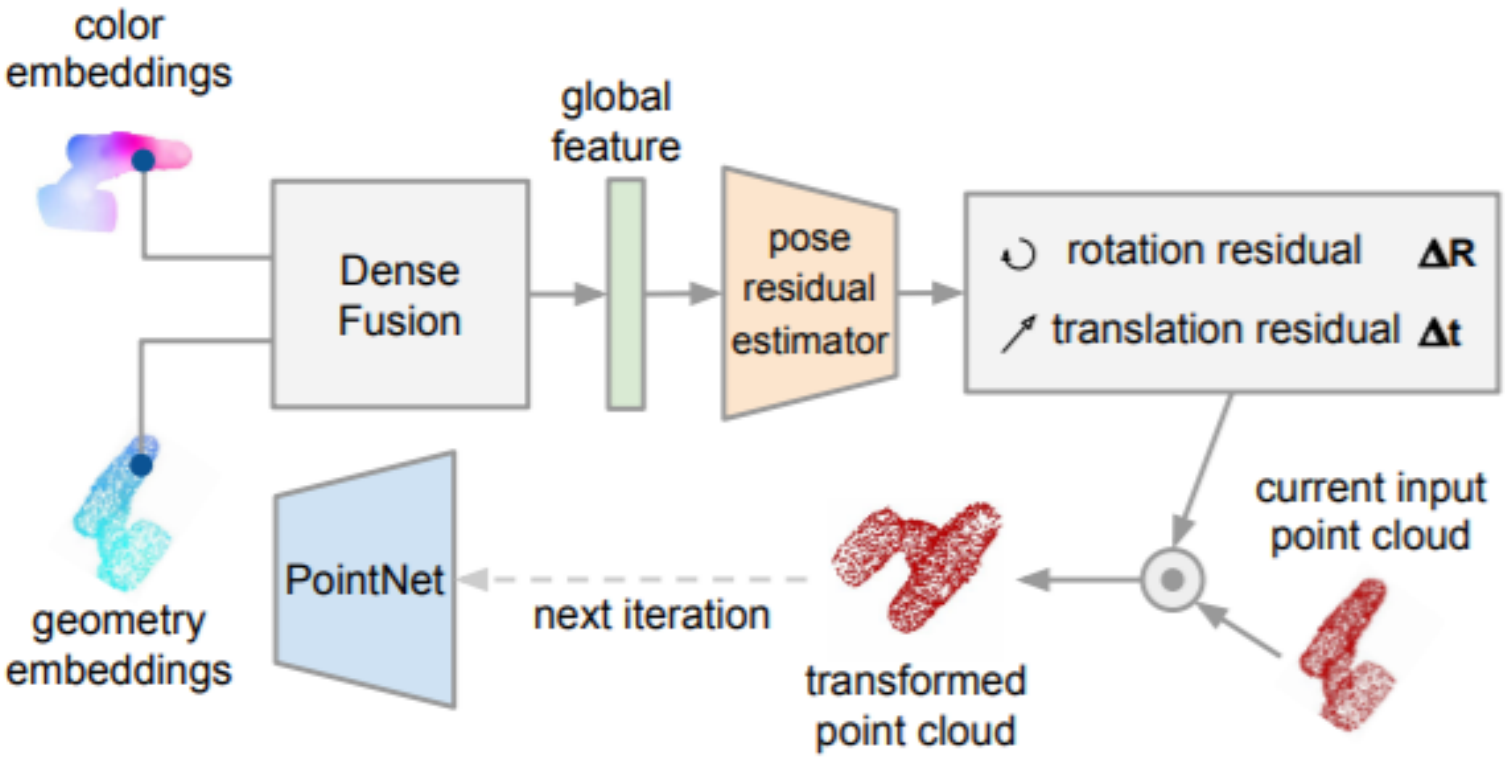
$$L_i^p = \frac{1}{M} \sum_j ||(Rx_j + t) - (\hat{R}_i x_j + \hat{t}_i)|| \qquad (1)$$

$$L_i^p = \frac{1}{M} \sum_j \min_{0<k<M} ||(Rx_j + t) - (\hat{R}_i x_k + \hat{t}_i)|| \qquad (2)$$

$$L = \frac{1}{N} \sum_i (L_i^p c_i - w \log(c_i)), \qquad (3)$$

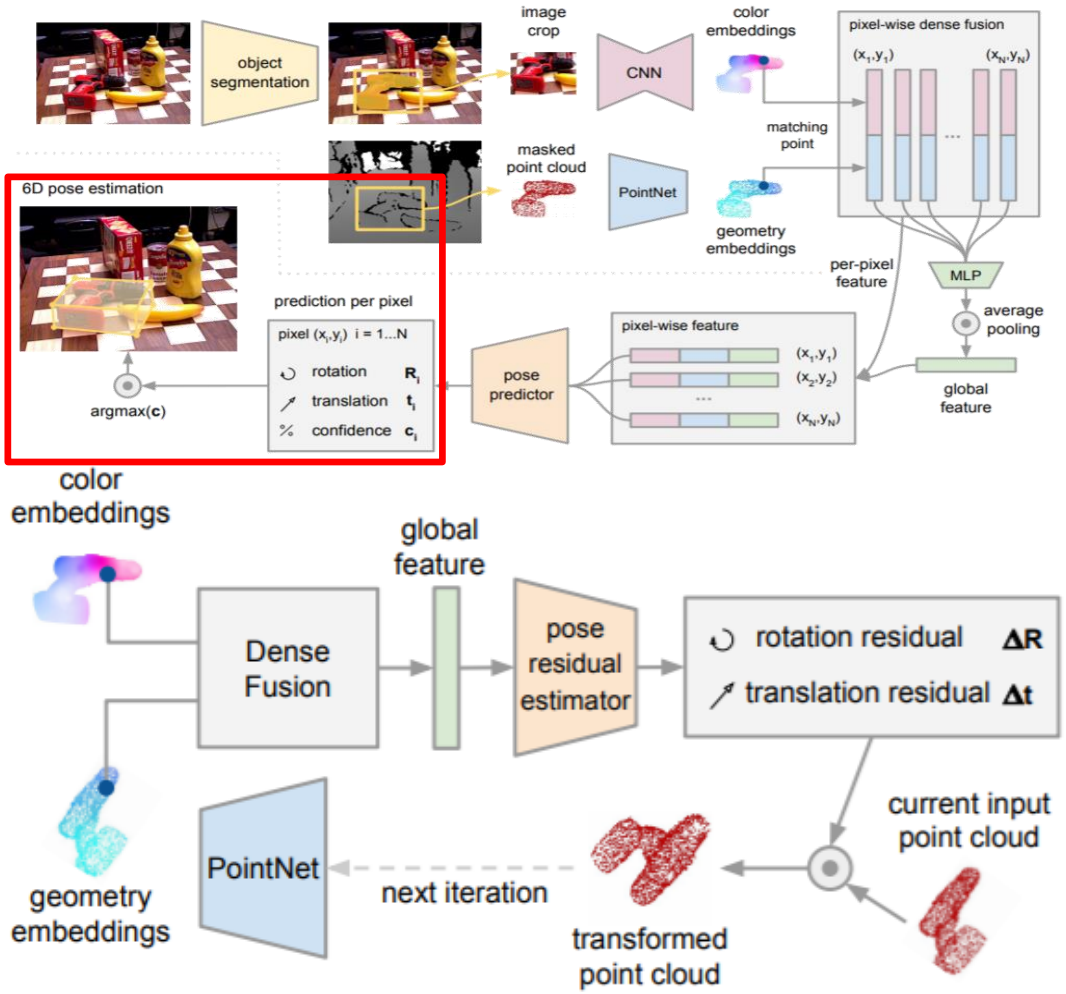
# 3. Model

3.6 iterative Refinement



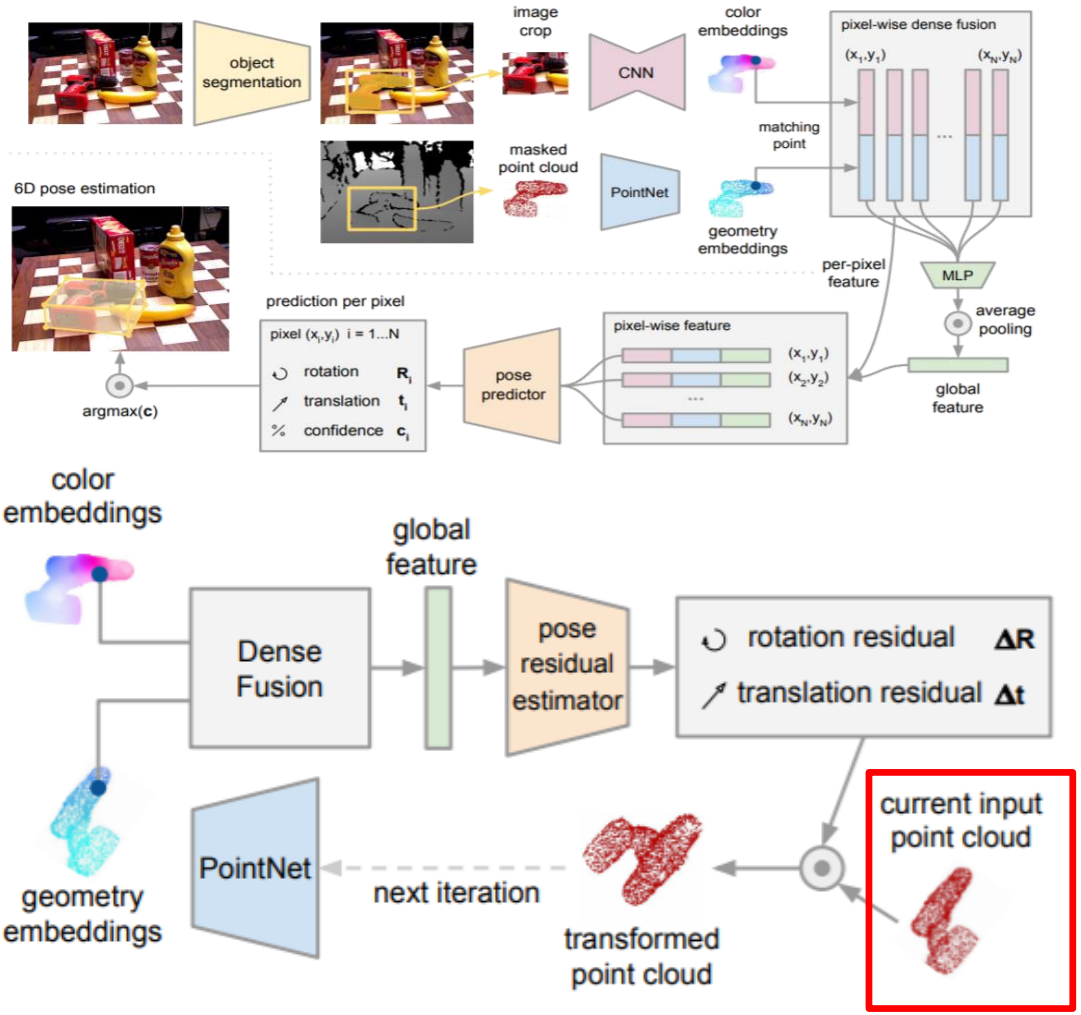
# 3. Model

## 3.6 iterative Refinement



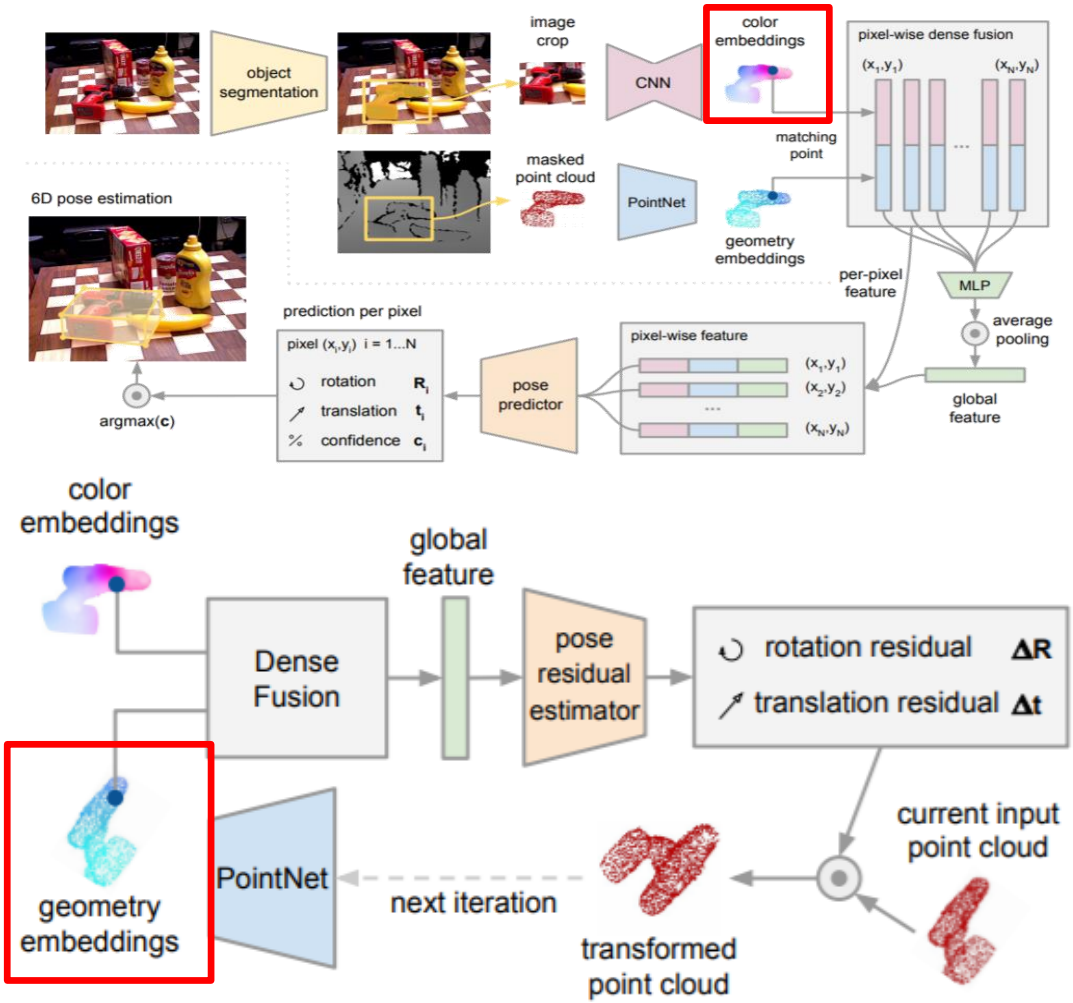
# 3. Model

## 3.6 iterative Refinement



# 3. Model

## 3.6 iterative Refinement



# 4. Experiments

1) how does the dense fusion network compare to naïve global fusion by concatenation

2) Is the dense fusion and prediction scheme robust to heavy occlusion and segmentation errors?

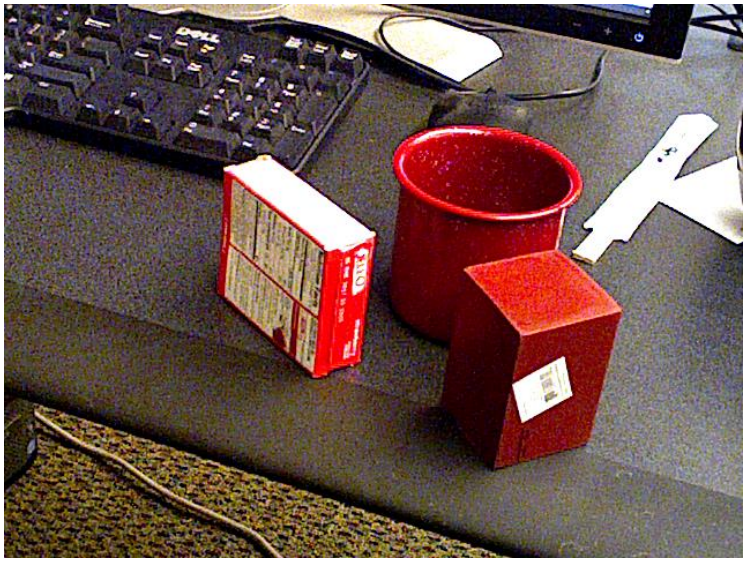
3) Does the iterative refinement module improve the final pose estimation

4) Is our method robust and efficient enough for downstream tasks such as robotic grasping ?

# 4. Experiments

## 4.1. Datasets

**YCB Video Dataset**



**LineMod**

**YCB Dataset 21개의 물체가 다양한 shape과 texture를 가지고 있음.**

**Linemod dataset : 13개의 low texture를 가지고 있는 물체**

# 4. Experiments

## 4.2. Metrics

We use two metrics to report on the YCB-Video Datset.

The avg closet point distance **ADD-S**(poseCNN) is an ambiguity invariant pose error metric which takes care of both symmetric and non-symmetric objects into an overall evaluation.

We report the area under the **ADD-S curve (AUC) following threshold of AUC to be 0.1m**. We also report the percentage of **ADD-S smaller than 2cm(<2cm)**, which measures the predictions under the minimum tolerance for robot manipulation.

For the LINEMod dataset, we use the Average distance of model points (ADD) for non symmetric objects and ADD-S for the two symmetric objects

# 4. Experiments

## 4.3. Architectures

We compare four model variants that showcase the effectiveness of our design choices.

**PointFusion** uses a CNN to extract a fixed-size feature vector and fuse by directly concatenating the image feature with the geometry feature. The rest of the network is similar to our architecture.

**PoseCNN +ICP**

**Ours(single)** uses our dense fusion network, but instead of performing per-point prediction, it only outputs a single prediction using the global feature vector.

**Ours(per-pixel)** performs per-pixel prediction based on each densely fused feature.

**Ours (iterative)** is our complete model that uses the iterative refinement on top of ours(per pixel)

# 4. Experiments

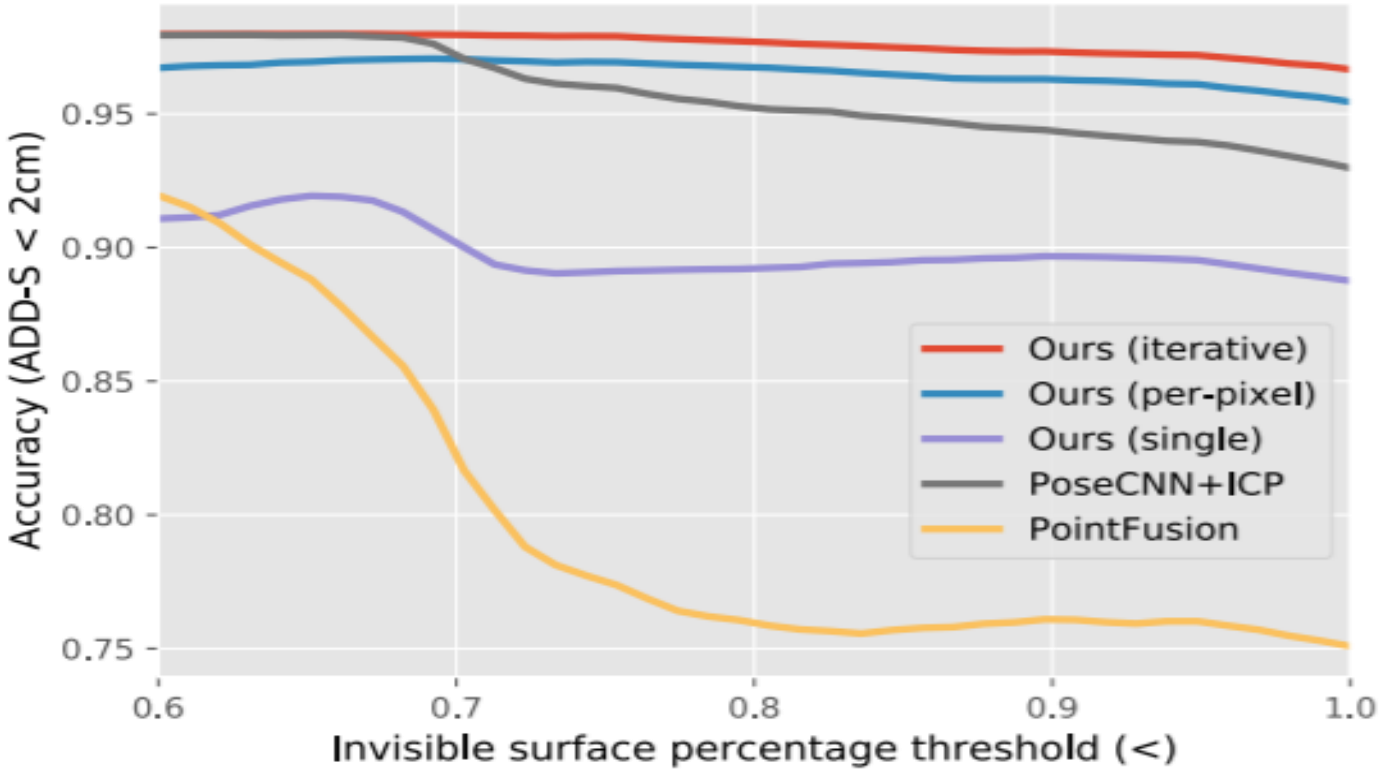
## 4.4. Evaluation on YCB-Video Dataset

Table 1. Quantitative evaluation of 6D pose (ADD-S[40]) on YCB-Video Dataset. Objects with bold name are symmetric.

| | PointFusion [41] | | PoseCNN+ICP [40] | | Ours (single) | | Ours (per-pixel) | | Ours (iterative) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | AUC | <2cm | AUC | <2cm | AUC | <2cm | AUC | <2cm | AUC | <2cm |
| 002_master_chef_can | 90.9 | 99.8 | 95.8 | 100.0 | 93.9 | 100.0 | 95.2 | 100.0 | **96.4** | 100.0 |
| 003_cracker_box | 80.5 | 62.6 | 92.7 | 91.6 | 90.8 | 98.4 | 92.5 | 99.3 | **95.5** | **99.5** |
| 004_sugar_box | 90.4 | 95.4 | **98.2** | 100.0 | 94.4 | 99.2 | 95.1 | 100.0 | 97.5 | 100.0 |
| 005_tomato_soup_can | 91.9 | 96.9 | 94.5 | 96.9 | 92.9 | 96.7 | 93.7 | 96.9 | **94.6** | 96.9 |
| 006_mustard_bottle | 88.5 | 84.0 | **98.6** | 100.0 | 91.2 | 97.8 | 95.9 | 100.0 | 97.2 | 100.0 |
| 007_tuna_fish_can | 93.8 | 99.8 | **97.1** | 100.0 | 94.9 | 100.0 | 94.9 | 100.0 | 96.6 | 100.0 |
| 008_pudding_box | 87.5 | 96.7 | **97.9** | 100.0 | 88.3 | 97.2 | 94.7 | 100.0 | 96.5 | 100.0 |
| 009_gelatin_box | 95.0 | 100.0 | **98.8** | 100.0 | 95.4 | 100.0 | 95.8 | 100.0 | 98.1 | 100.0 |
| 010_potted_meat_can | 86.4 | 88.5 | **92.7** | **93.6** | 87.3 | 91.4 | 90.1 | 93.1 | 91.3 | 93.1 |
| 011_banana | 84.7 | 70.5 | **97.1** | 99.7 | 84.6 | 62.0 | 91.5 | 93.9 | 96.6 | **100.0** |
| 019_pitcher_base | 85.5 | 79.8 | **97.8** | 100.0 | 86.9 | 80.9 | 94.6 | 100.0 | 97.1 | 100.0 |
| 021_bleach_cleanser | 81.0 | 65.0 | **96.9** | 99.4 | 91.6 | 98.2 | 94.3 | 99.8 | 95.8 | **100.0** |
| **024_bowl** | 75.7 | 24.1 | 81.0 | 54.9 | 83.4 | 55.4 | 86.6 | 69.5 | **88.2** | **98.8** |
| 025_mug | 94.2 | 99.8 | 95.0 | 99.8 | 90.3 | 94.7 | 95.5 | **100.0** | **97.1** | **100.0** |
| 035_power_drill | 71.5 | 22.8 | **98.2** | **99.6** | 83.1 | 64.2 | 92.4 | 97.1 | 96.0 | 98.7 |
| **036_wood_block** | 68.1 | 18.2 | 87.6 | 80.2 | 81.7 | 76.0 | 85.5 | 93.4 | **89.7** | **94.6** |
| 037_scissors | 76.7 | 35.9 | 91.7 | 95.6 | 83.6 | 75.1 | 96.4 | **100.0** | **95.2** | **100.0** |
| 040_large_marker | 87.9 | 80.4 | 97.2 | 99.7 | 91.2 | 88.6 | 94.7 | 99.2 | **97.5** | **100.0** |
| **051_large_clamp** | 65.9 | 50.0 | **75.2** | 74.9 | 70.5 | 77.1 | 71.6 | 78.5 | 72.9 | **79.2** |
| **052_extra_large_clamp** | 60.4 | 20.1 | 64.4 | 48.8 | 66.4 | 50.2 | 69.0 | 69.5 | **69.8** | **76.3** |
| **061_foam_brick** | 91.8 | 100.0 | **97.2** | 100.0 | 92.1 | 100.0 | 92.4 | 100.0 | 92.5 | 100.0 |
| MEAN | 83.9 | 74.1 | 93.0 | 93.2 | 88.2 | 87.9 | 91.2 | 95.3 | **93.1** | **96.8** |

# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset



# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset

| PoseCNN+ICP [40] | | | | Ours | | | |
|---|---|---|---|---|---|---|---|
| Seg | PE | ICP | ALL | Seg | PE | Refine | ALL |
| 0.03 | 0.17 | 10.4 | 10.6 | 0.03 | 0.02 | 0.01 | 0.06 |

# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset

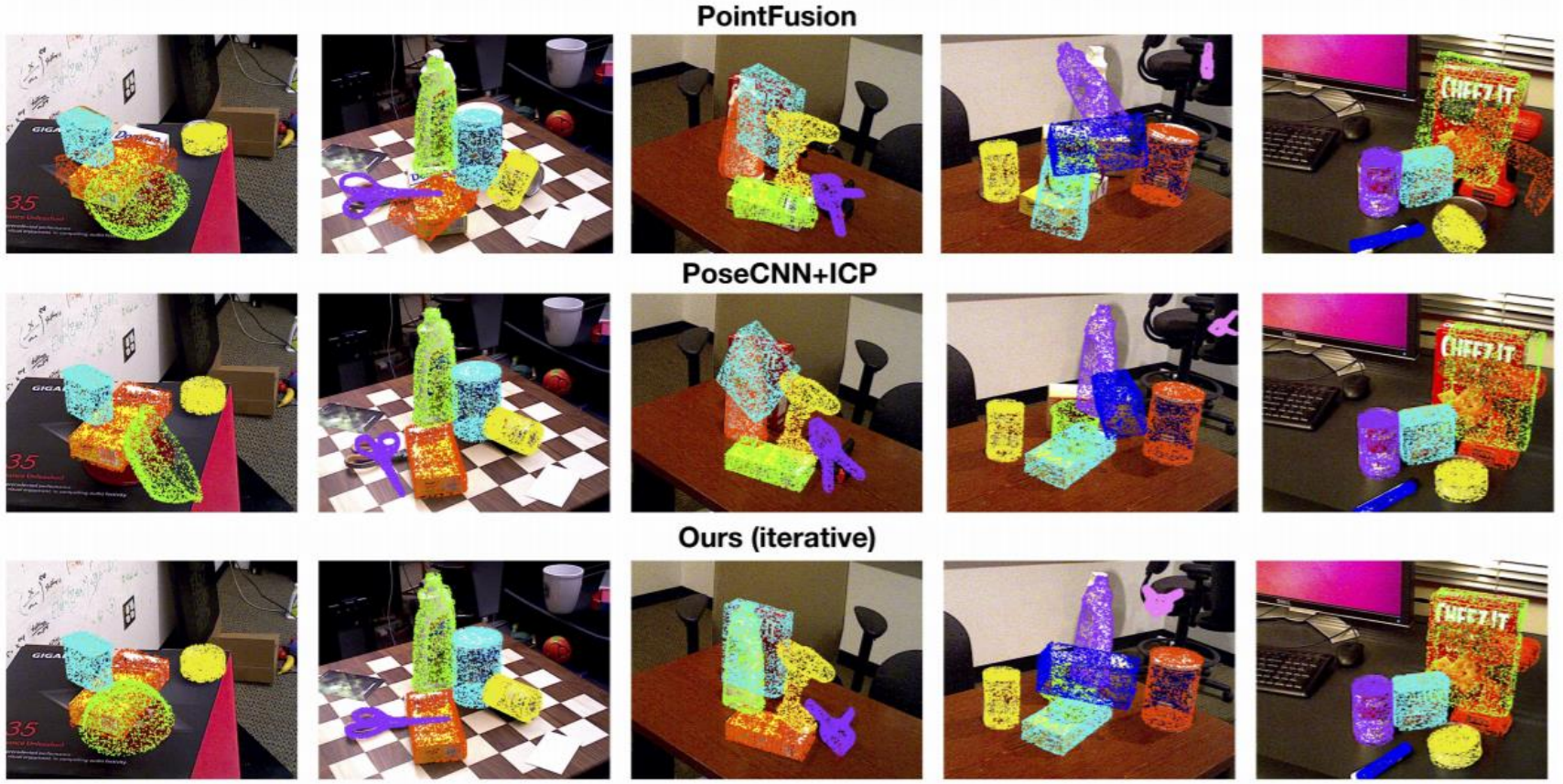



Figure 4. **Qualitative results on the YCB-Video Dataset.** All three methods shown here are tested with the same segmentation masks as in PoseCNN. Each object point cloud in different color are transformed with the predicted pose and then projected to the 2D image frame. The first two rows are former RGB-D methods and the last row is our approach with dense fusion and iterative refinement (2 iterations).

# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset

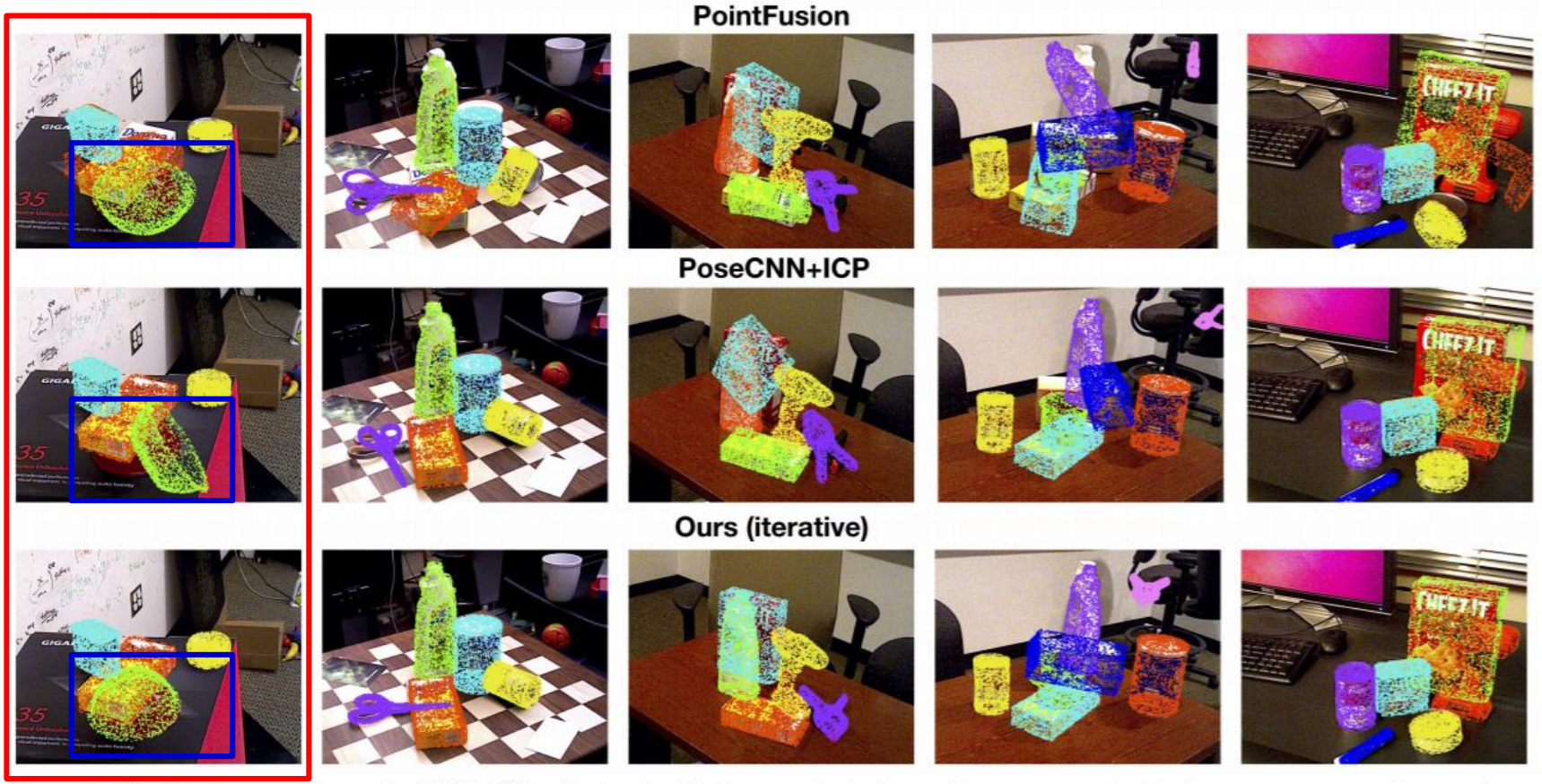



Figure 4. **Qualitative results on the YCB-Video Dataset.** All three methods shown here are tested with the same segmentation masks as in PoseCNN. Each object point cloud in different color are transformed with the predicted pose and then projected to the 2D image frame. The first two rows are former RGB-D methods and the last row is our approach with dense fusion and iterative refinement (2 iterations).

# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset

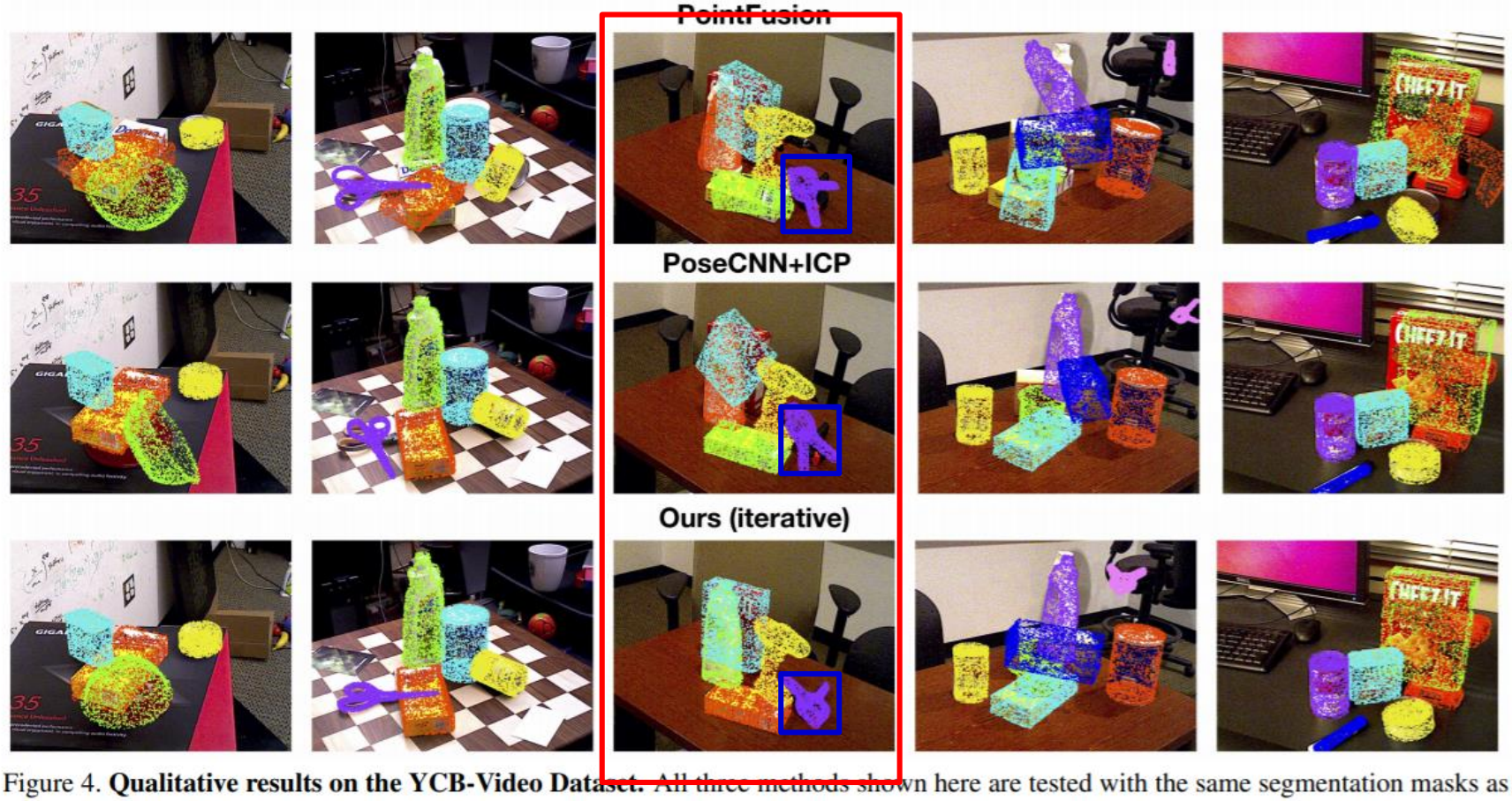



Figure 4. **Qualitative results on the YCB-Video Dataset.** All three methods shown here are tested with the same segmentation masks as in PoseCNN. Each object point cloud in different color are transformed with the predicted pose and then projected to the 2D image frame. The first two rows are former RGB-D methods and the last row is our approach with dense fusion and iterative refinement (2 iterations).

# 4. Experiments

## 4.4. Evaluation on YCB-Video Dataset

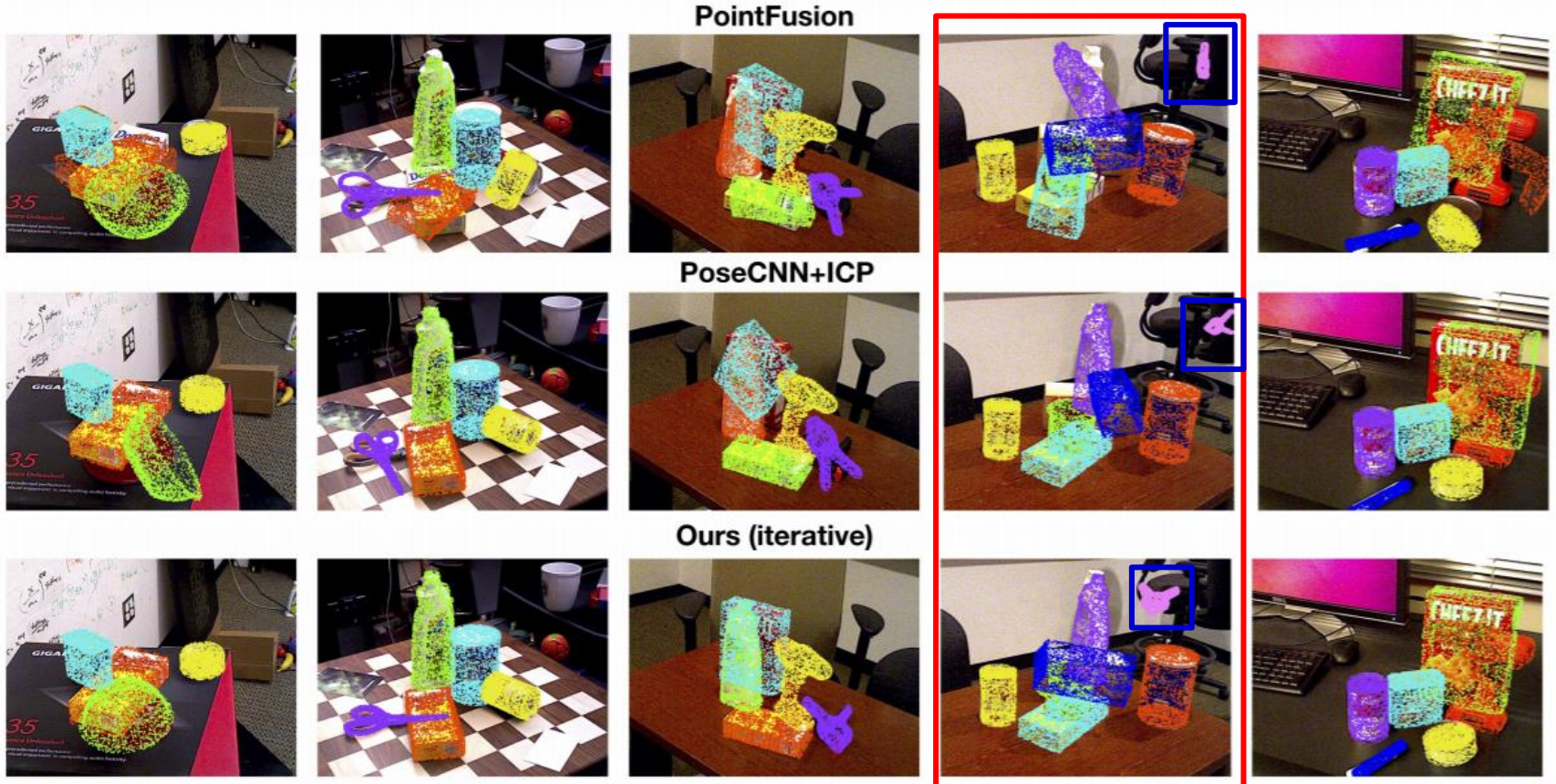



Figure 4. **Qualitative results on the YCB-Video Dataset.** All three methods shown here are tested with the same segmentation masks as in PoseCNN. Each object point cloud in different color are transformed with the predicted pose and then projected to the 2D image frame. The first two rows are former RGB-D methods and the last row is our approach with dense fusion and iterative refinement (2 iterations).

# 4. Experiments

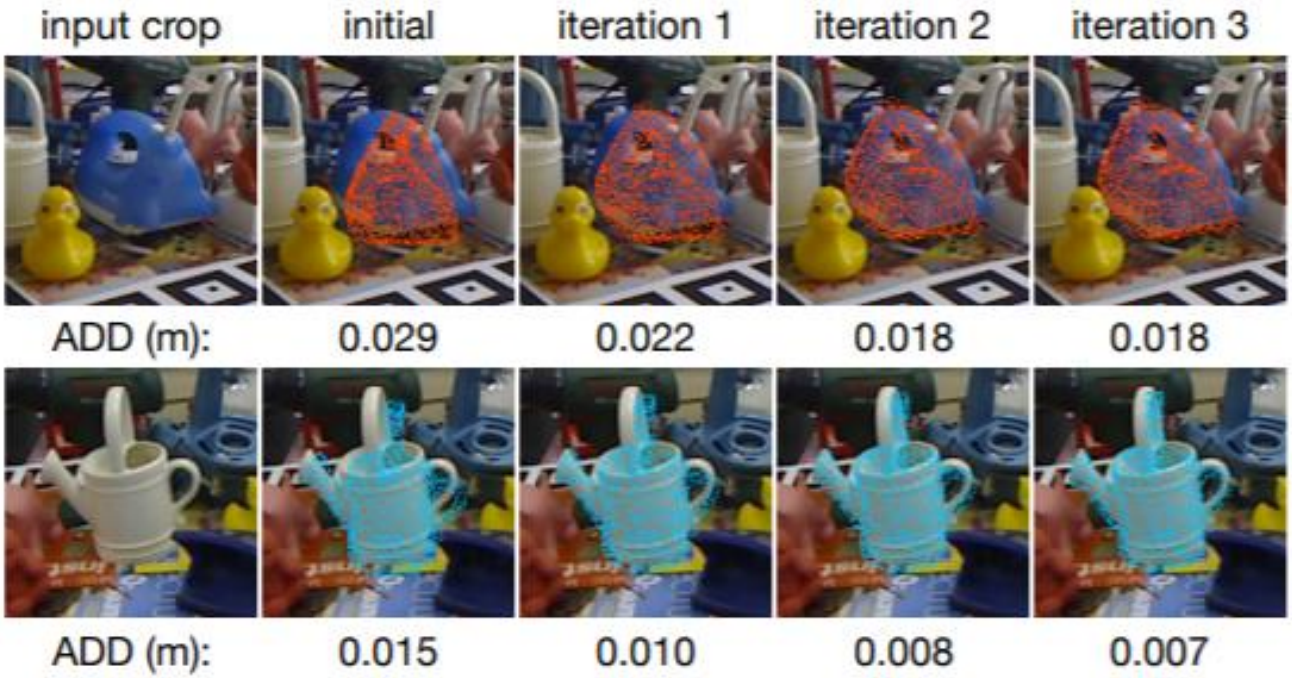
## 4.5 Evaluation on LineMOD Dataset

Table 2. Quantitative evaluation of 6D pose (ADD[13]) on the LineMOD dataset. Objects with bold name are symmet

| | RGB | | RGB-D | | | | |
|---|---|---|---|---|---|---|---|
| | BB8 [24] w ref. | PoseCNN +DeepIM [17, 40] | Implicit [30]+ICP | SSD-6D [14]+ICP | PointFusion [41] | Ours (per-pixel) | Ours (iterative) |
| ape | 40.4 | 77.0 | 20.6 | 65 | 70.4 | 79.5 | 92.3 |
| bench vi. | 91.8 | 97.5 | 64.3 | 80 | 80.7 | 84.2 | 93.2 |
| camera | 55.7 | 93.5 | 63.2 | 78 | 60.8 | 76.5 | 94.4 |
| can | 64.1 | 96.5 | 76.1 | 86 | 61.1 | 86.6 | 93.1 |
| cat | 62.6 | 82.1 | 72.0 | 70 | 79.1 | 88.8 | 96.5 |
| driller | 74.4 | 95.0 | 41.6 | 73 | 47.3 | 77.7 | 87.0 |
| duck | 44.3 | 77.7 | 32.4 | 66 | 63.0 | 76.3 | 92.3 |
| **eggbox** | 57.8 | 97.1 | 98.6 | 100 | 99.9 | 99.9 | 99.8 |
| **glue** | 41.2 | 99.4 | 96.4 | 100 | 99.3 | 99.4 | 100.0 |
| hole p. | 67.2 | 52.8 | 49.9 | 49 | 71.8 | 79.0 | 92.1 |
| iron | 84.7 | 98.3 | 63.1 | 78 | 83.2 | 92.1 | 97.0 |
| lamp | 76.5 | 97.5 | 91.7 | 73 | 62.3 | 92.3 | 95.3 |
| phone | 54.0 | 87.7 | 71.0 | 79 | 78.8 | 88.0 | 92.8 |
| MEAN | 62.7 | 88.6 | 64.7 | 79 | 73.7 | 86.2 | 94.3 |

# 4. Experiments

## 4.5 Evaluation on LineMOD Dataset



# 5. Conclusion

1. We presented a **novel approach to estimating 6D** poses of known objects from RGB-D images.

2. Our approach fuses a dense representation of features that **include color and depth information based on the confidence of their predictions.**

3. With this dense fusion approach, our method **outperforms** previous approaches in several datasets, and is significantly more **robust against occlusions**.

4. Additionally, we demonstrated that a robot can use our proposed approach to grasp and manipulate objects

# End