PVN3D : A Deep Point-wise 3D keypoints Voting Network for 6DoF Pose Estimation CVPR-2020

Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, Jian Sun

KIST

송명하

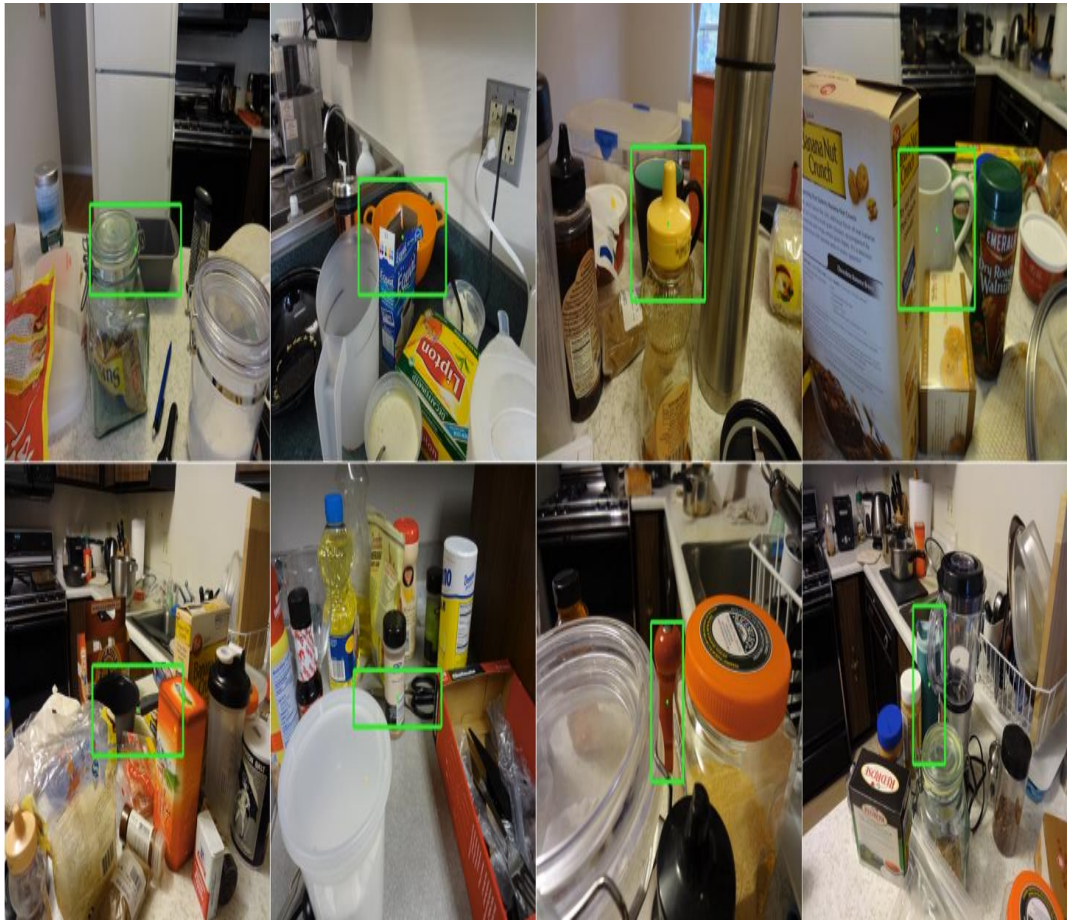
Content

1. Introduction

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

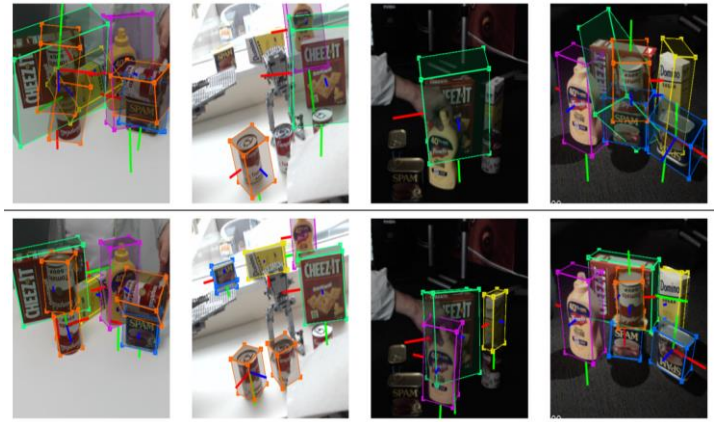
https://www.independent.co.uk/extras/indybest/gadgets-tech/video-games-consoles/augmented-reality-games-ar-android-ios-apple-google-play-tech-a8506831.html

1. Introduction



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

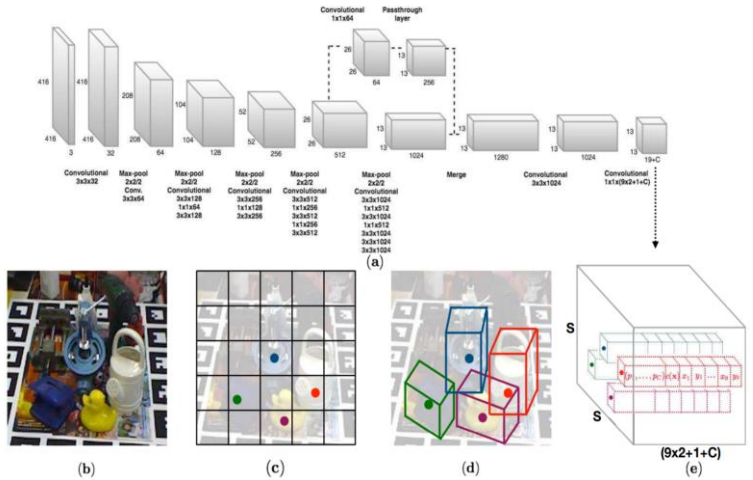
https://www.photoreview.com.au/tips/shooting/how-to-control-image-noise/



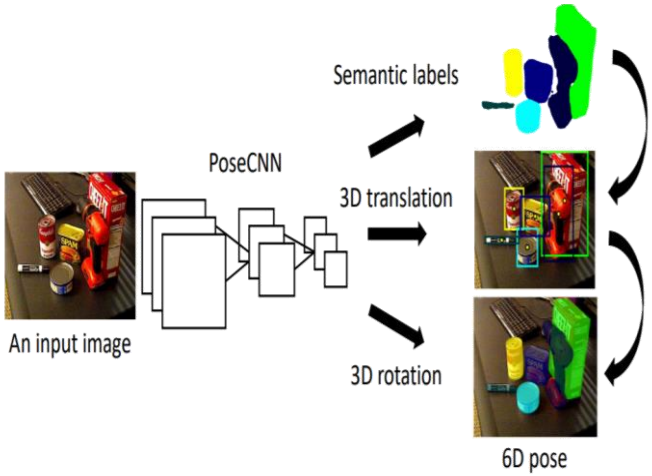
Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects

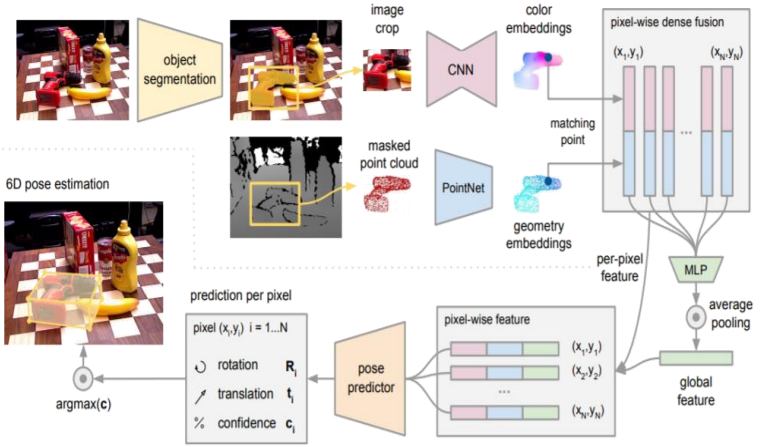
1. Introduction

Key Points



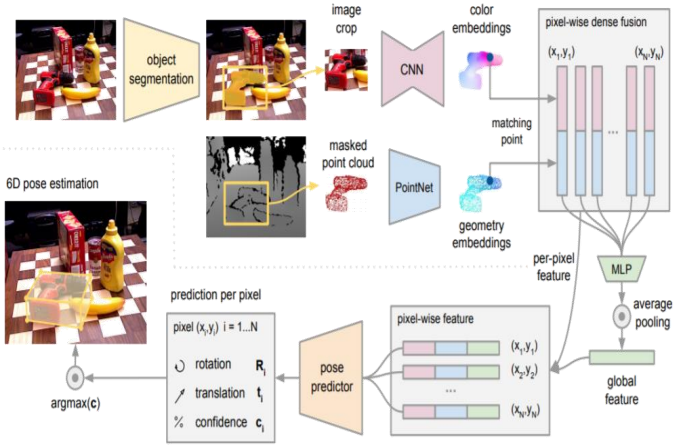
Direct Regression

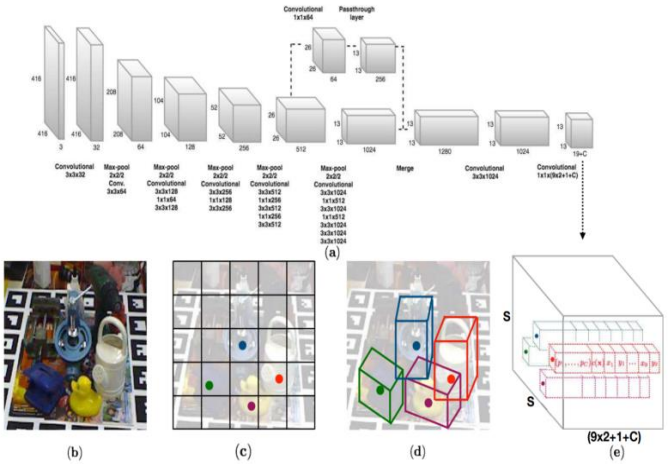




1. Introduction

PVN3D = KeyPoints + Direct Regression





Network :Direct Regression

Output :KeyPoints

1. Introduction

Contribution

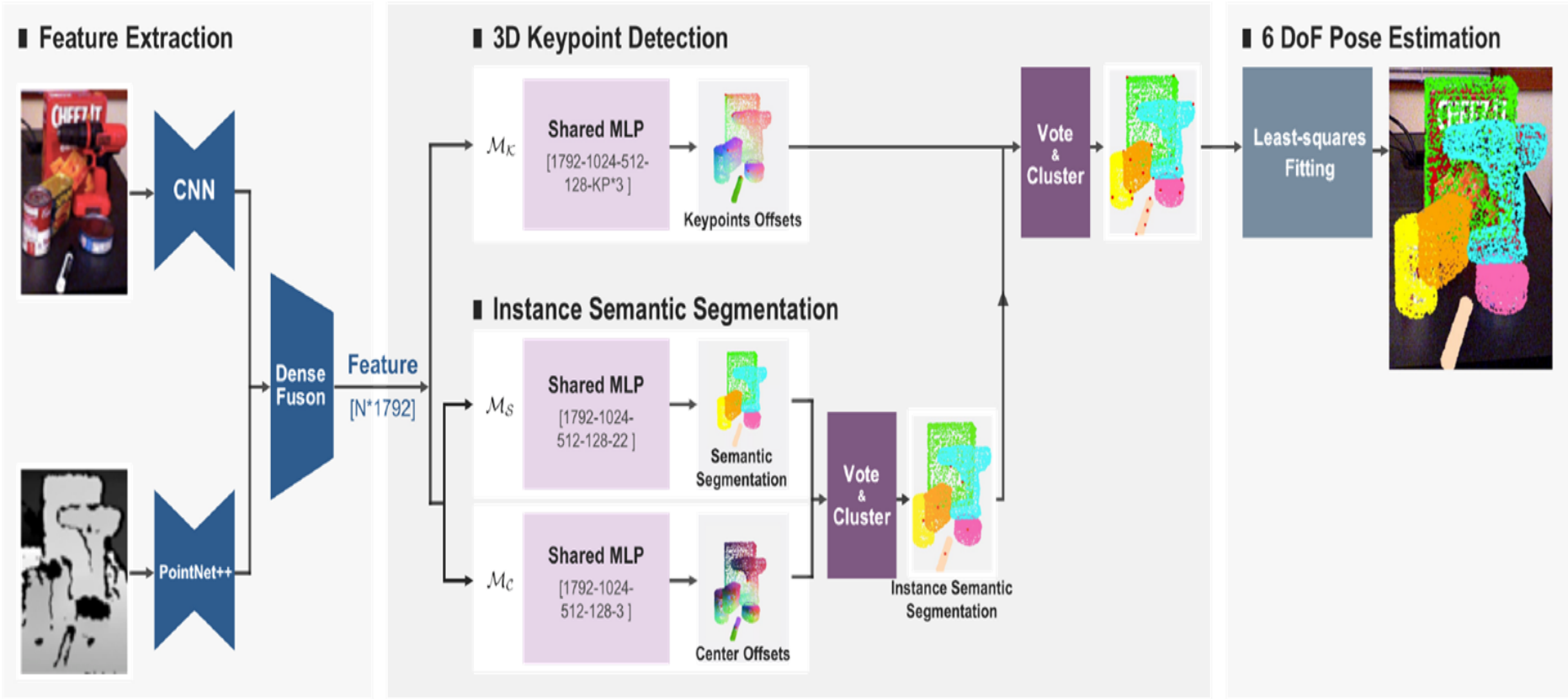
1. A novel deep 3D keypoints Hough voting network with instance semantic segmentation for 6DoF Pose Estimation of single RGB-D image

2. State-of-the art 6DoF pose estimation performance on YCB and LineMOD datasets.

3. An in-depth analysis of our 3D-keypoint-based method and comparison with previous approaches, demonstrating that 3D-keypoint is a key factor to boost performance for 6DoF pose estimation. We also show that jointly training 3D-keypoint and semantic segmentation can further improve the performance.

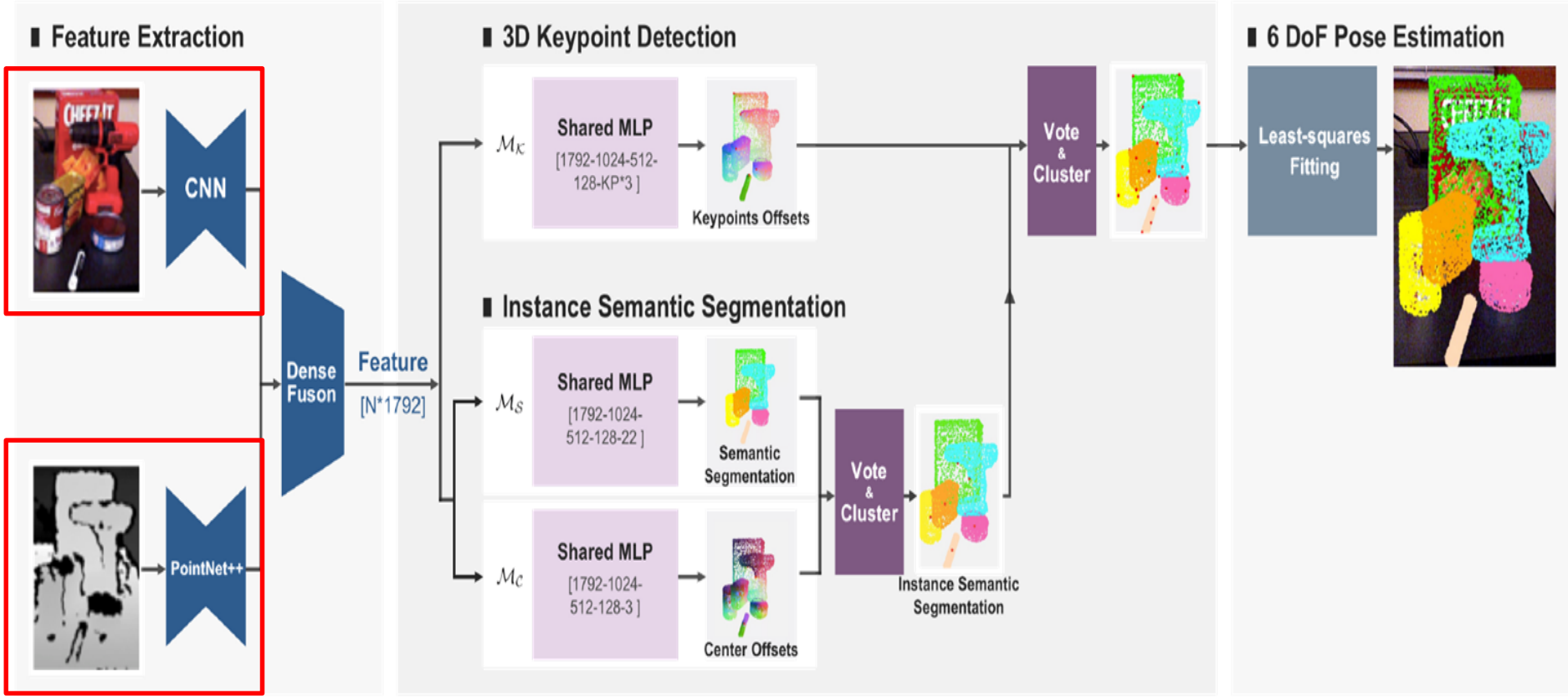
3. Model

3.1 Architecture Overview

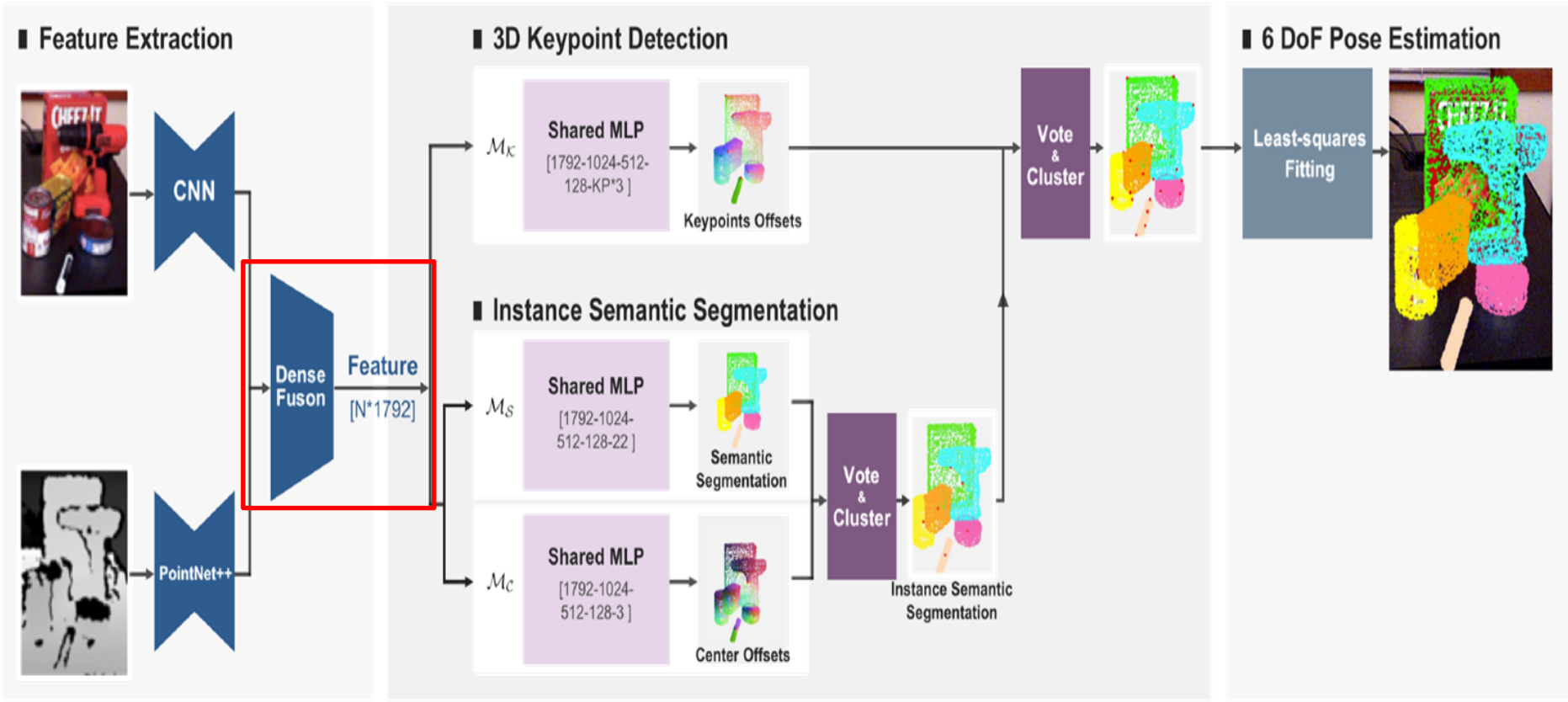


3. Model

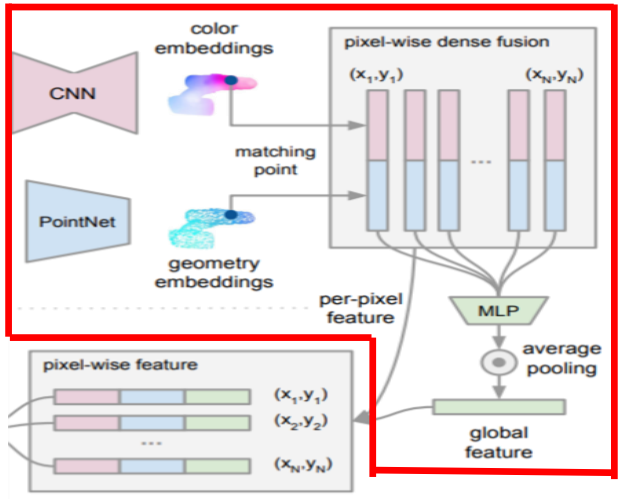
3.1 Architecture Overview



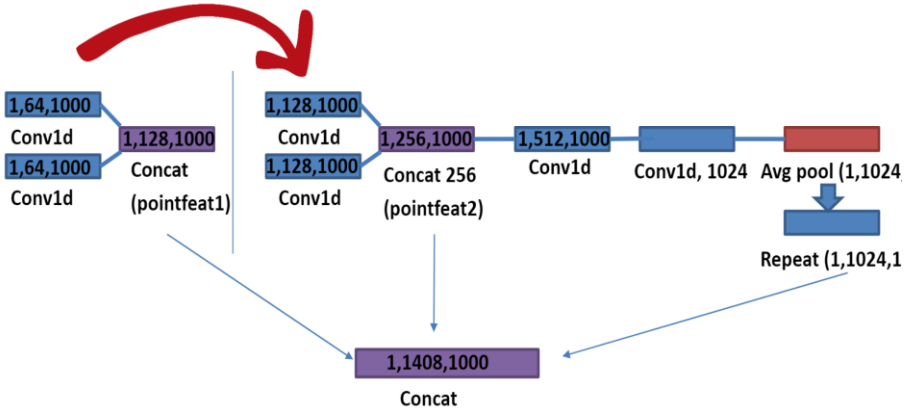
3. Model

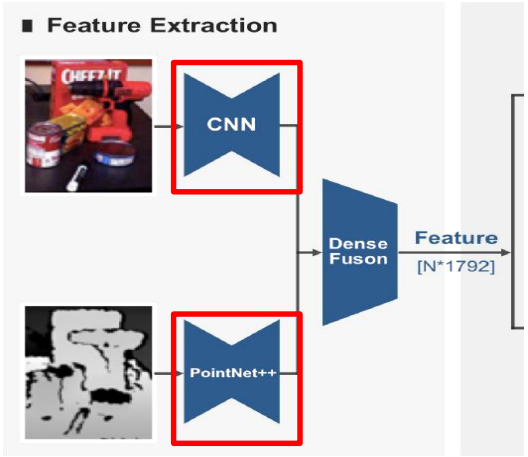

Feature Extraction
CNN
PointNet++
Dense Fuson
Feature
[N*1792]
3D Keypoint Detection
$\mathcal{M}_K$
Shared MLP
[1792-1024-512-128-KP*3]
Keypoints Offsets
Instance Semantic Segmentation
$\mathcal{M}_S$
Shared MLP
[1792-1024-512-128-22]
Semantic Segmentation
$\mathcal{M}_C$
Shared MLP
[1792-1024-512-128-3]
Center Offsets
Vote & Cluster
Instance Semantic Segmentation
Vote & Cluster
6 DoF Pose Estimation
Least-squares Fitting

3. Model

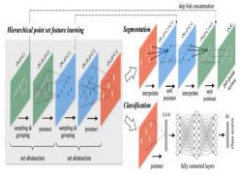


RESNET18

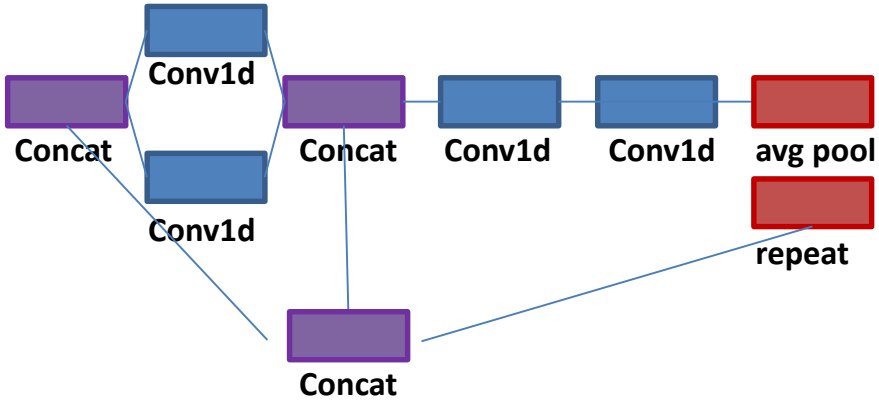




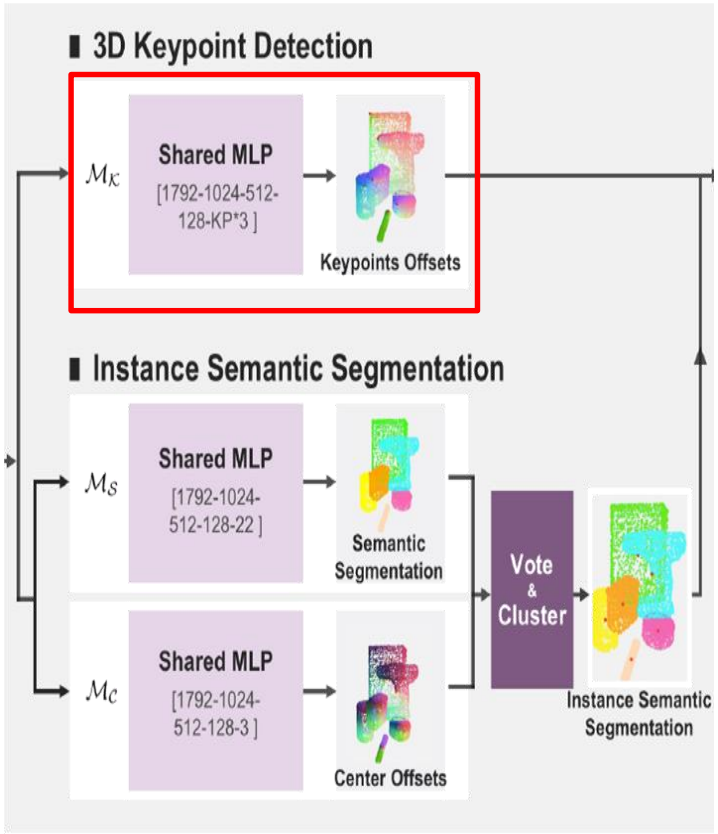
RESNET34



Pointnet++



3. Model



$\{p_i\}_{i=1}^{N}$ **Visible Point**

$\{kp_j\}_{j=1}^{M}$ **Keypoints**

x_i **3D coordinates**

f_i **features**

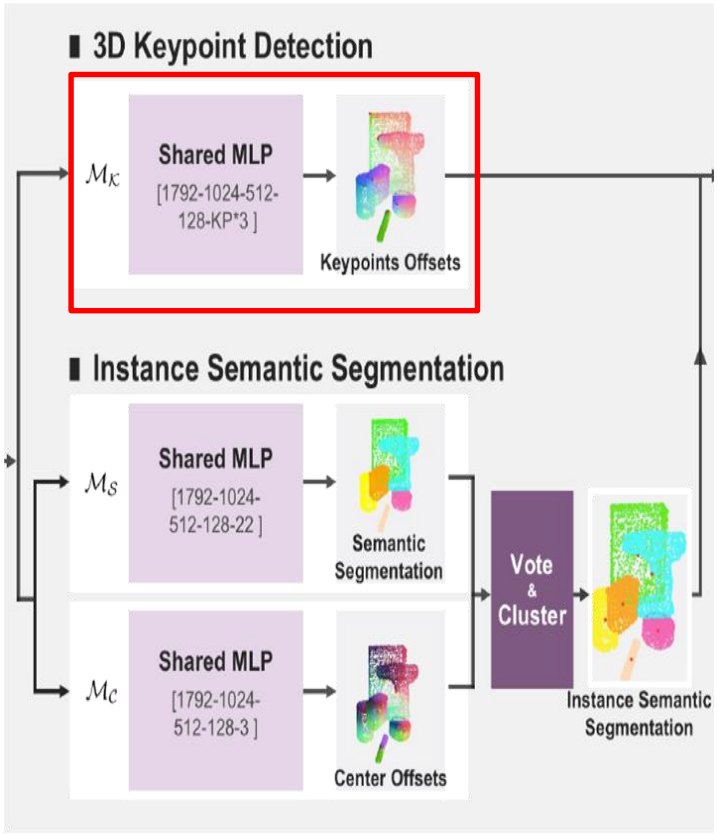
$$p_i = [x_i; f_i]$$

$\{of_i^j\}_{j=1}^{M}$ **Translation offsets**

$$kp_j = [y_j]$$

$vkp_i^j = x_i + of_i^j$. **Voted keypoints**

3. Model



L1 Loss

$$L_{\text{keypoints}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}||of_i^j - of_i^{j*}||\mathbb{I}(p_i \in I) \quad (1)$$

Fusion Feature 1792

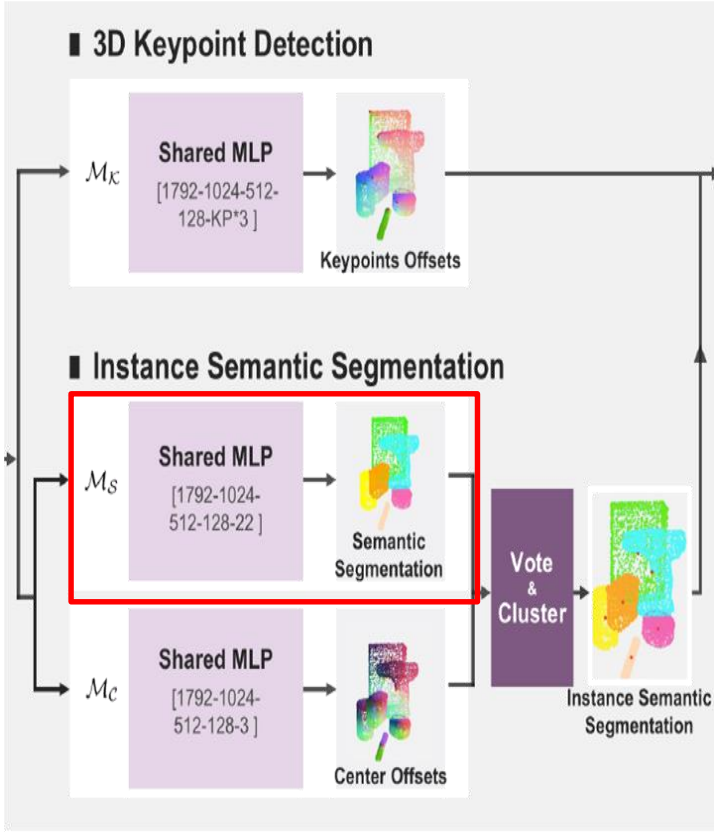
Conv1d 1024

Conv1d 512

Conv1d 256

Conv1d num*3

3. Model



Focal Loss

$$L_{semantic} = -\alpha(1-q_i)^{\gamma} log(q_i)$$
$$where \quad q_i = c_i \cdot l_i$$

with α the α-balance parameter, γ the focusing parameter, c_i the predicted confidence for the i_{th} point belongs to each class and l_i the one-hot representation of ground true class label.

Fusion Feature 1792

Conv1d 1024

Conv1d 512

Conv1d 128

Conv1d num_c

Loss

Focal Loss(Focal Loss for Dense Object Detection – 2017)

Dice Loss

IoU

Loss

Focal Loss(Focal Loss for Dense Object Detection – 2017)

$$\mathrm{FL}(p_t) = -(1-p_t)^{\gamma}\log(p_t).$$

잘 찾은 것에 대해서는 loss를 적게 줘서 loss갱신을 거의 하지 못하게 하고,
잘 못 찾은 것은 loss를 크게 하자.

```
math.log(0.9)  -0.10536051565782628
gamma 0 FL  0.10536051565782628
gamma 1 FL  0.010536051565782625
gamma 3 FL  0.00010536051565782622
```

```
math.log(0.1)  -2.302585092994045
gamma 0 FL  2.302585092994045
gamma 1 FL  2.072326583694641
gamma 3 FL  1.6785845327926594
```

https://ufris.tistory.com/17

Loss

Dice Loss / IoU

$$\mathrm{DC} = \frac{2TP}{2TP + FP + FN} = \frac{2|X \cap Y|}{|X| + |Y|}$$

$$\mathrm{IoU} = \frac{TP}{TP + FP + FN} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|}$$

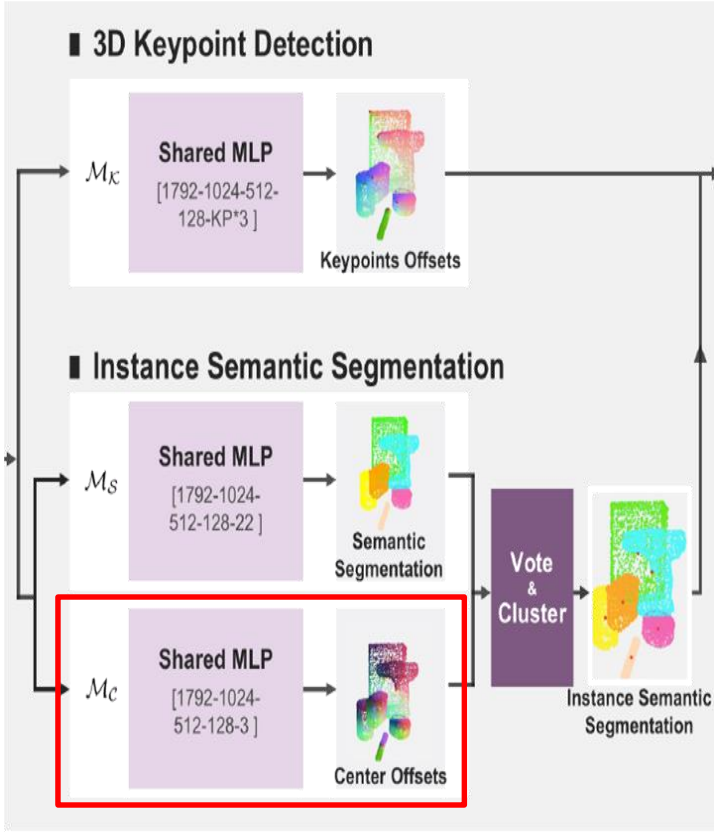
	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

$$\mathrm{DL}\left(p, \hat{p}\right) = 1 - \mathrm{DC}$$

http://blog.naver.com/PostView.nhn?blogId=kmkim1222&logNo=220106232149&parentCategoryNo=&categoryNo=24&viewDate=&isShowPopularPosts=false&from=postView

https://lars76.github.io/neural-networks/object-detection/losses-for-segmentation/

3. Model



L1 Loss

$$L_{\text{center}} = \frac{1}{N}\sum_{i=1}^{N} ||\Delta x_i - \Delta x_i^*|| \mathbb{I}(p_i \in I) \qquad (3)$$

Fusion Feature 1792

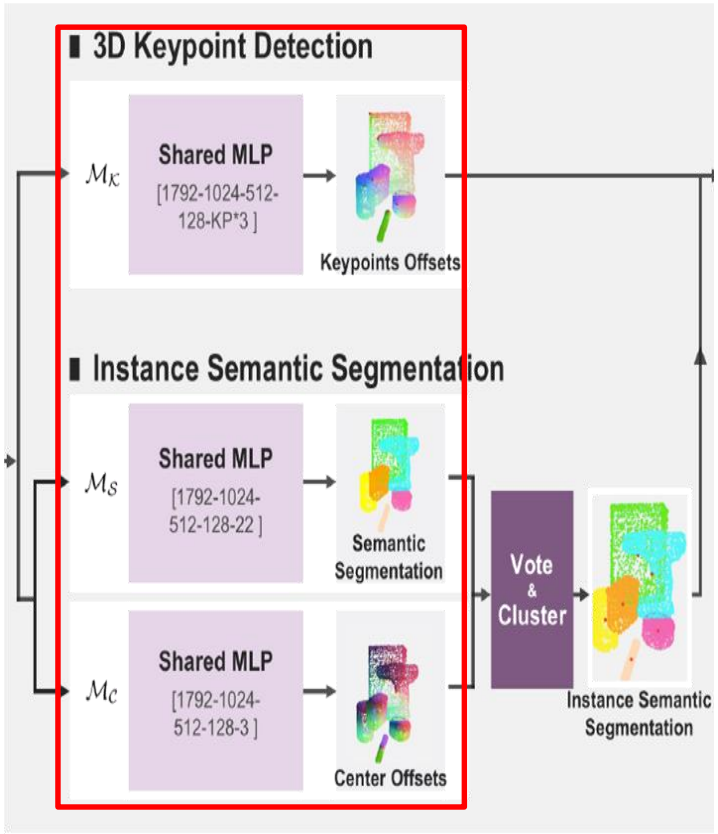
Conv1d 1024

Conv1d 512

Conv1d 128

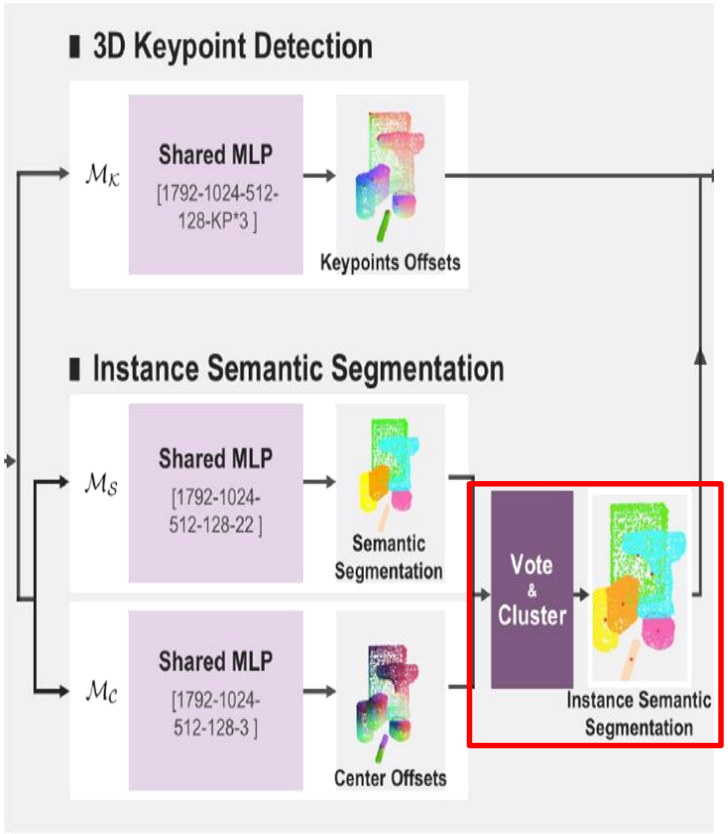
Conv1d 3

3. Model

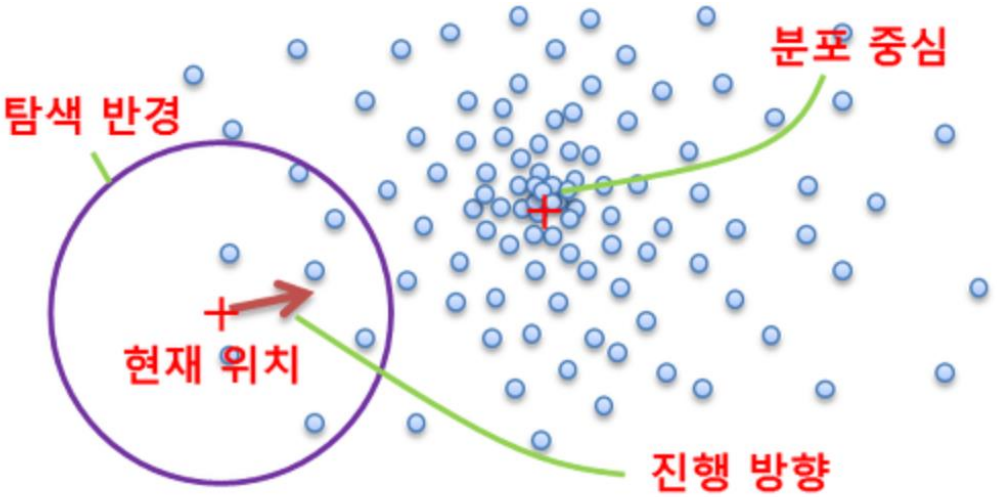


$$L_{\text{multi-task}} = \lambda_1 L_{\text{keypoints}} + \lambda_2 L_{\text{semantic}} + \lambda_3 L_{\text{center}} \quad (4)$$

3. Model



Mean Shift



https://darkpgmr.tistory.com/64

3. Model – Training and Implementation

Sample : N points

λ 값은 1모두 1로 고정.

Keypoint Selection

대부분의 알고리즘이 8개의 bounding box corner 하지만 가상의 corne라 에러가 커짐.

그래서 대신에 우리는 farthest point sampling (FPS) algorithm 으로 keypoints를 mesh에서 select함.

Least-Squares Fitting.

하나의 M개의 keypoints는 카메라 coordinate system이고 하나는 object coordinate system이라 그것 두개를 (R,t) pose paramete로 least-squares fitting algorithm을 이용해서 R과 t를 찾는다.

$$L_{\text{least-squares}} = \sum_{j=1}^{M} ||kp_j - (R \cdot kp_j^{'} + t)||^2 \qquad (5)$$

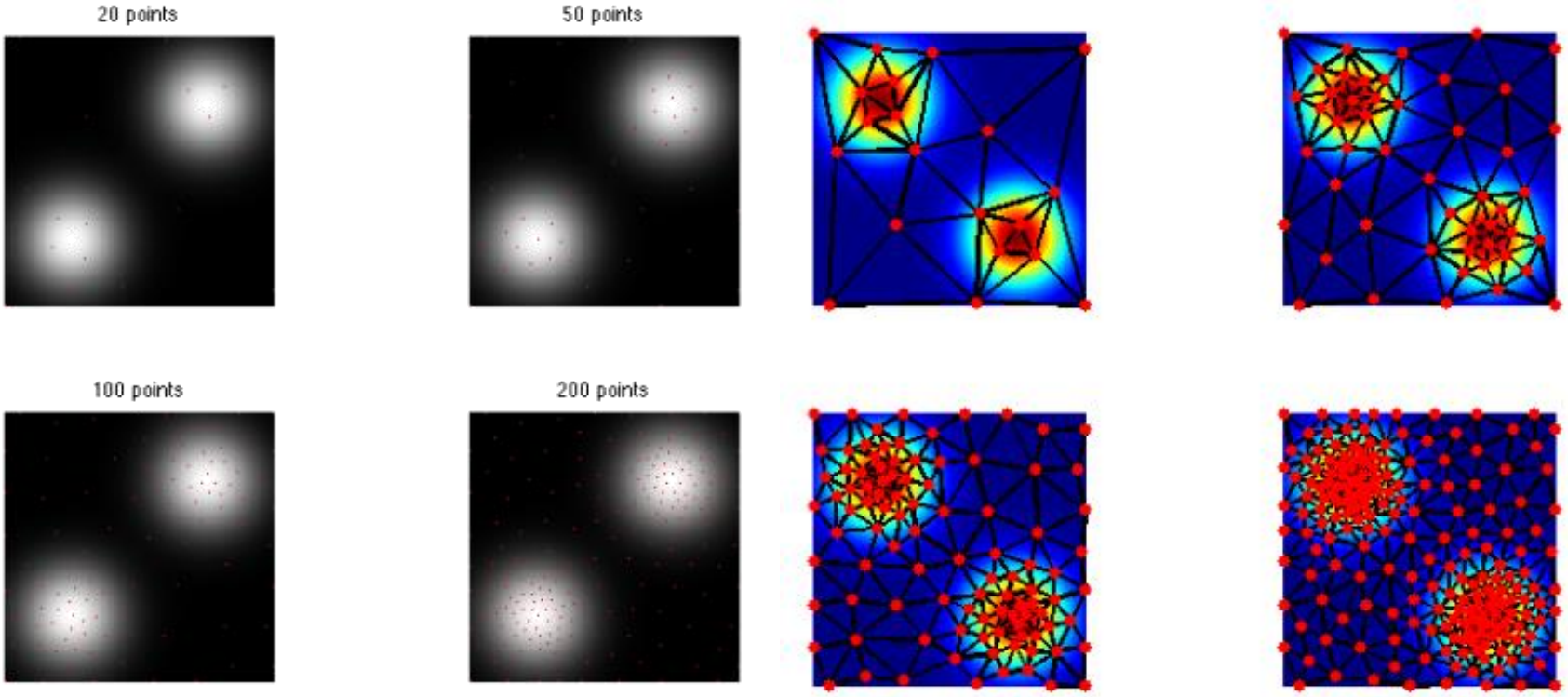
3. Model – Training and Implementation

Farthest point sampling (FPS) algorithm

First. We initialize the keypoint set by adding the object center.

Then we repeatedly find a point on the object surface, which is farthest to the current keypoint set, and add it top the set until the size of the set reaches K.

Considering both accuracy and efficient, we suggest K = 8 according the the experiment results.



3. Model – Training and Implementation

Sample : N points

λ 값은 1모두 1로 고정.

Keypoint Selection

대부분의 알고리즘이 8개의 bounding box corner 하지만 가상의 corne라 에러가 커짐.

그래서 대신에 우리는 farthest point sampling (FPS) algorithm 으로 keypoints를 mesh에서 select함.

Least-Squares Fitting.

하나의 M개의 keypoints는 카메라 coordinate system이고 하나는 object coordinate system이라 그것 두개를 (R,t) pose paramete로 least-squares fitting algorithm을 이용해서 R과 t를 찾는다.

$$L_{\text{least-squares}} = \sum_{j=1}^{M} ||kp_j - (R \cdot kp_j^{'} + t)||^2 \qquad (5)$$

4. Experiments – Datasets

YCB Video Dataset.

LineMOD Dataset.

4. Experiments – Evaluation Metrics

ADD and ADD-S metric

$$\text{ADD} = \frac{1}{m} \sum_{x \in \mathcal{O}} ||(Rx + t) - (R^* x + t^*)|| \qquad (6)$$

$$\text{ADD-S} = \frac{1}{m} \sum_{x_1 \in \mathcal{O}} \min_{x_2 \in \mathcal{O}} ||(Rx_1 + t) - (R^* x_2 + t^*)|| \qquad (7)$$

4. Experiments – Evaluiation on YCB-Video & LineMOD Dataset

	Without Iterative Refinement						With Iterative Refinement					
	PoseCNN[52]		DF(per-pixel)[50]		PVN3D		PoseCNN+ICP[52]		DF(iterative)[50]		PVN3D+ICP	
	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)
002_master_chef_can	83.9	50.2	95.3	70.7	96.0	**80.5**	95.8	68.1	**96.4**	73.2	95.2	79.3
003_cracker_box	76.9	53.1	92.5	86.9	**96.1**	**94.8**	92.7	83.4	95.8	94.1	94.4	91.5
004_sugar_box	84.2	68.4	95.1	90.8	97.4	96.3	**98.2**	**97.1**	97.6	96.5	97.9	96.9
005_tomato_soup_can	81.0	66.2	93.8	84.7	**96.2**	88.5	94.5	81.8	94.5	85.5	95.9	**89.0**
006_mustard_bottle	90.4	81.0	95.8	90.9	97.5	96.2	**98.6**	**98.0**	97.3	94.7	98.3	97.9
007_tuna_fish_can	88.0	70.7	95.7	79.6	96.0	89.3	**97.1**	83.9	**97.1**	81.9	96.7	**90.7**
008_pudding_box	79.1	62.7	94.3	89.3	97.1	95.7	97.9	96.6	96.0	93.3	**98.2**	**97.1**
009_gelatin_box	87.2	75.2	97.2	95.8	97.7	96.1	98.8	98.1	98.0	96.7	**98.8**	**98.3**
010_potted_meat_can	78.5	59.5	89.3	79.6	93.3	**88.6**	92.7	83.5	90.7	83.6	**93.8**	87.9
011_banana	86.0	72.3	90.0	76.7	96.6	93.7	97.1	91.9	96.2	83.3	**98.2**	**96.0**
019_pitcher_base	77.0	53.3	93.6	87.1	97.4	96.5	**97.8**	96.9	97.5	96.9	97.6	**96.9**
021_bleach_cleanser	71.6	50.3	94.4	87.5	96.0	93.2	96.9	92.5	95.9	89.9	**97.2**	**95.9**
024_bowl	69.6	69.6	86.0	86.0	90.2	90.2	81.0	81.0	89.5	89.5	**92.8**	**92.8**
025_mug	78.2	58.5	95.3	83.8	97.6	95.4	94.9	81.1	96.7	88.9	**97.7**	**96.0**
035_power_drill	72.7	55.3	92.1	83.7	96.7	95.1	**98.2**	**97.7**	96.0	92.7	97.1	95.7
036_wood_block	64.3	64.3	89.5	89.5	90.4	90.4	87.6	87.6	**92.8**	**92.8**	91.1	91.1
037_scissors	56.9	35.8	90.1	77.4	**96.7**	**92.7**	91.7	78.4	92.0	77.9	95.0	87.2
040_large_marker	71.7	58.3	95.1	89.1	96.7	91.8	97.2	85.3	97.6	**93.0**	**98.1**	91.6
051_large_clamp	50.2	50.2	71.5	71.5	93.6	93.6	75.2	75.2	72.5	72.5	**95.6**	**95.6**
052_extra_large_clamp	44.1	44.1	70.2	70.2	88.4	88.4	64.4	64.4	69.9	69.9	**90.5**	**90.5**
061_foam_brick	88.0	88.0	92.2	92.2	96.8	96.8	97.2	97.2	92.0	92.0	**98.2**	**98.2**
ALL	75.8	59.9	91.2	82.9	95.5	91.8	93.0	85.4	93.2	86.1	**96.1**	**92.3**

Table 1. Quantitative evaluation of 6D Pose (ADD-S AUC [52], ADD(S) AUC [19]) on the YCB-Video Dataset. Symmetric objects' names are in bold.

Iterative refinement과정없이 성능 압도함.

4. Experiments – Evaluiation on YCB-Video & LineMOD Dataset

	Without Iterative Refinement						With Iterative Refinement					
	PoseCNN[52]		DF(per-pixel)[50]		PVN3D		PoseCNN+ICP[52]		DF(iterative)[50]		PVN3D+ICP	
	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)
002_master_chef_can	83.9	50.2	95.3	70.7	96.0	**80.5**	95.8	68.1	**96.4**	73.2	95.2	79.3
003_cracker_box	76.9	53.1	92.5	86.9	**96.1**	**94.8**	92.7	83.4	95.8	94.1	94.4	91.5
004_sugar_box	84.2	68.4	95.1	90.8	97.4	96.3	**98.2**	**97.1**	97.6	96.5	97.9	96.9
005_tomato_soup_can	81.0	66.2	93.8	84.7	**96.2**	88.5	94.5	81.8	94.5	85.5	95.9	**89.0**
006_mustard_bottle	90.4	81.0	95.8	90.9	97.5	96.2	**98.6**	**98.0**	97.3	94.7	98.3	97.9
007_tuna_fish_can	88.0	70.7	95.7	79.6	96.0	89.3	**97.1**	83.9	**97.1**	81.9	96.7	**90.7**
008_pudding_box	79.1	62.7	94.3	89.3	97.1	95.7	97.9	96.6	96.0	93.3	**98.2**	**97.1**
009_gelatin_box	87.2	75.2	97.2	95.8	97.7	96.1	98.8	98.1	98.0	96.7	**98.8**	**98.3**
010_potted_meat_can	78.5	59.5	89.3	79.6	93.3	**88.6**	92.7	83.5	90.7	83.6	**93.8**	87.9
011_banana	86.0	72.3	90.0	76.7	96.6	93.7	97.1	91.9	96.2	83.3	**98.2**	**96.0**
019_pitcher_base	77.0	53.3	93.6	87.1	97.4	96.5	**97.8**	96.9	97.5	96.9	97.6	**96.9**
021_bleach_cleanser	71.6	50.3	94.4	87.5	96.0	93.2	96.9	92.5	95.9	89.9	**97.2**	**95.9**
024_bowl	69.6	69.6	86.0	86.0	90.2	90.2	81.0	81.0	89.5	89.5	**92.8**	**92.8**
025_mug	78.2	58.5	95.3	83.8	97.6	95.4	94.9	81.1	96.7	88.9	**97.7**	**96.0**
035_power_drill	72.7	55.3	92.1	83.7	96.7	95.1	**98.2**	**97.7**	96.0	92.7	97.1	95.7
036_wood_block	64.3	64.3	89.5	89.5	90.4	90.4	87.6	87.6	**92.8**	**92.8**	91.1	91.1
037_scissors	56.9	35.8	90.1	77.4	**96.7**	**92.7**	91.7	78.4	92.0	77.9	95.0	87.2
040_large_marker	71.7	58.3	95.1	89.1	96.7	91.8	97.2	85.3	97.6	**93.0**	**98.1**	91.6
051_large_clamp	50.2	50.2	71.5	71.5	93.6	93.6	75.2	75.2	72.5	72.5	**95.6**	**95.6**
052_extra_large_clamp	44.1	44.1	70.2	70.2	88.4	88.4	64.4	64.4	69.9	69.9	**90.5**	**90.5**
061_foam_brick	88.0	88.0	92.2	92.2	96.8	96.8	97.2	97.2	92.0	92.0	**98.2**	**98.2**
ALL	75.8	59.9	91.2	82.9	95.5	91.8	93.0	85.4	93.2	86.1	**96.1**	**92.3**

Table 1. Quantitative evaluation of 6D Pose (ADD-S AUC [52], ADD(S) AUC [19]) on the YCB-Video Dataset. Symmetric objects' names are in bold.

Iterative refinement 넣으면 PVN3D + ICP하면 우리는 더 좋아짐.

그리고 얘들은 extra-large clamp랑 large clamp 잘함.

4. Experiments – Evaluiation on YCB-Video & LineMOD Dataset

Robust to Occlusion Scenes.

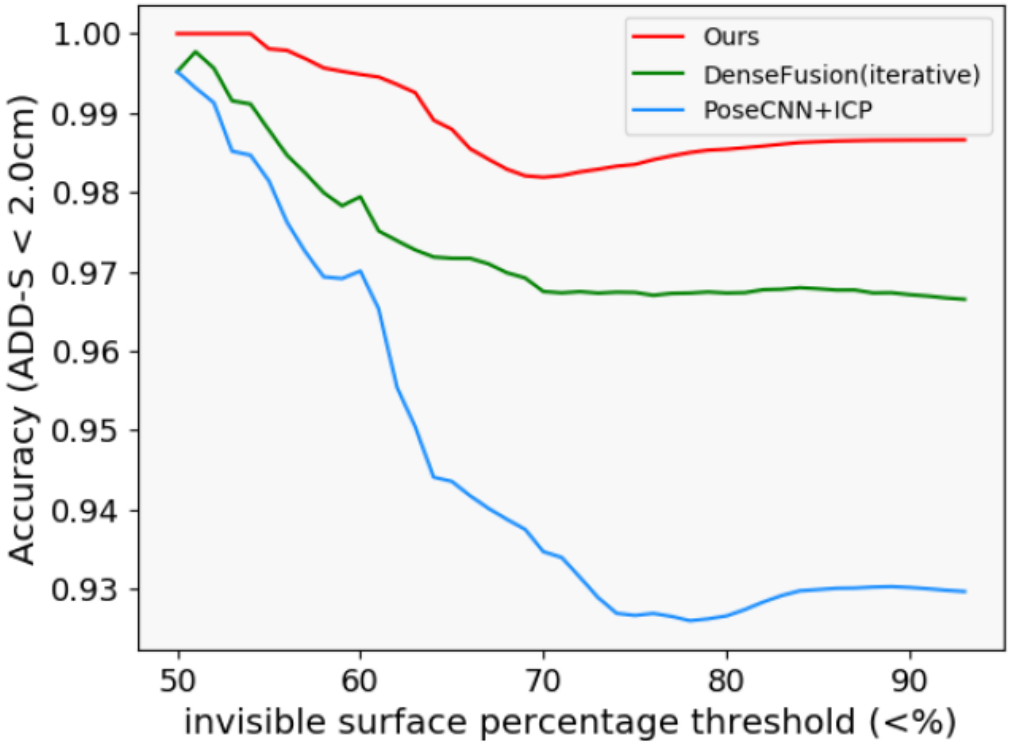


Figure 4. Performance of different approaches under increasing levels of occlusion on the YCB-Video dataset.

4. Experiments – Ablation Study

Comparisons to Directly Regressing Pose.

	DF(RT)[50]	DF(3D KP)[50]	Ours(RT)	Ours(2D KPC)	Ours(2D KP)	PVNet[37]	Ours(Corr)	Ours(3D KP)
ADD-S	92.2	93.1	92.8	78.2	81.8	-	92.8	**95.5**
ADD(S)	86.9	87.9	87.3	73.8	77.2	73.4	88.1	**91.8**

Table 4. Quantitative evaluation of 6D Poses on the YCB-Video dataset with different formulations. All with our predicted segmentation.

Mk를 direct로 R과 T를 regression하는 방식으로 바꿔서 해봤음.
(DenseFusion방식)

4. Experiments – Ablation Study

Comparisons to Directly Regressing Pose.

	DF(RT)[50]	DF(3D KP)[50]	Ours(RT)	Ours(2D KPC)	Ours(2D KP)	PVNet[37]	Ours(Corr)	Ours(3D KP)
ADD-S	92.2	93.1	92.8	78.2	81.8	-	92.8	**95.5**
ADD(S)	86.9	87.9	87.3	73.8	77.2	73.4	88.1	**91.8**

Table 4. Quantitative evaluation of 6D Poses on the YCB-Video dataset with different formulations. All with our predicted segmentation.

DenseFusion network를 keypoints를 predict하는 방식으로 바꿔서 해봄.

3D keypoint offset 검색하는 space가 rotation space의 비선형성보다 작기 때문에 신경망이 학습하기 쉽고 일반화 할수 있기때문이라 생각.

4. Experiments – Ablation Study

Comparisons to 2D Keypoints.

	DF(RT)[50]	DF(3D KP)[50]	Ours(RT)	Ours(2D KPC)	Ours(2D KP)	PVNet[37]	Ours(Corr)	Ours(3D KP)
ADD-S	92.2	93.1	92.8	78.2	81.8	-	92.8	**95.5**
ADD(S)	86.9	87.9	87.3	73.8	77.2	73.4	88.1	**91.8**

Table 4. Quantitative evaluation of 6D Poses on the YCB-Video dataset with different formulations. All with our predicted segmentation.

3D keypoint를 2D keypoint 로 projection 시켜서 test(with PnP algorithm)

PnP Algorithm은 projection error를 최소하는 것이 목적.

Projection에서 자그마한 error는 3D pose상에서는 매우 큼.

4. Experiments – Ablation Study

Comparisons to 2D Keypoints.

	DF(RT)[50]	DF(3D KP)[50]	Ours(RT)	Ours(2D KPC)	Ours(2D KP)	PVNet[37]	Ours(Corr)	Ours(3D KP)
ADD-S	92.2	93.1	92.8	78.2	81.8	-	92.8	**95.5**
ADD(S)	86.9	87.9	87.3	73.8	77.2	73.4	88.1	**91.8**

Table 4. Quantitative evaluation of 6D Poses on the YCB-Video dataset with different formulations. All with our predicted segmentation.

Instance semantic segmentation 에서 2D center와 3D center point를 비교하기 위해서 3D Point를 2D instance semantic segmentation module로 구성해서 실험해 봄.

Mean shift를 이용해 clustering을 했는데 occlusio장면에서 center가 서로 가까이 있을 때 잘 안됨.

Heat map으로 keypoint하는 방법들 또는 vector voting하는 방법들은 keypoint overlap되는 문제가 있었음.

4. Experiments – Ablation Study

Effect of 3D keypoints Selection.

	VoteNet[38]	BBox 8	FPS 4	FPS 8	FPS 12
ADD-S	89.9	94.0	94.3	**95.5**	94.5
ADD(S)	85.1	90.2	90.5	**91.8**	90.7

Table 5. Effect of different keypoint selection methods of PVN3D. Results of VoteNet[38], another 3D bounding box detection approach are added as a simple baseline to compare with our BBox8.

Bounding box의 corner는 가상의 points라 실제 object point와 상이함.

그러므로 point based network는 bounding box point부근에서 context를 집계하기 어려움.

FPS는 network가 학습하기 좋게 잘 choice 됨.

4. Experiments – Ablation Study

Effect of Multi-task learning

	$\mathcal{M}_{\mathcal{K}}$ +MRC	$\mathcal{M}_{\mathcal{K}}$ +GT	$\mathcal{M}_{\mathcal{K},\mathcal{S}}$ +GT	$\mathcal{M}_{\mathcal{K},\mathcal{S},\mathcal{C}}$	$\mathcal{M}_{\mathcal{K},\mathcal{S},\mathcal{C}}$ +GT
ADD-S	93.5	94.8	95.2	95.5	**95.7**
ADD(S)	89.7	90.6	91.3	91.8	**91.9**

Table 6. Performance of PVN3D with different instance semantic segmentation on all objects in the YCB-Video dataset. $\mathcal{M}_{\mathcal{K}}$, $\mathcal{M}_{\mathcal{S}}$ and $\mathcal{M}_{\mathcal{C}}$ denote keypoint offset module, semantic segmentation and center point offset module of PVN3D respectively. +MRC and +GT denotes inference with segmentation result of Mask R-CNN and ground truth segmentation respectively.

Semantic segmentation 과 center voting module이 전체 학습을 boost시키는 것을 확인.

Segmentation 학습 시 서로 잘 구별하기 위해서 global, local feature를 잘 extract 한다고 생각.

4. Experiments – Ablation Study

Effect of Multi-task learning

	PoseCNN [52]	Mask R-CNN[15]	PVN3D ($\mathcal{M}_{\mathcal{S}}$)	PVN3D ($\mathcal{M}_{\mathcal{S},\mathcal{K}}$)	PVN3D ($\mathcal{M}_{\mathcal{S},\mathcal{K},\mathcal{C}}$)
large clamp	43.1	48.4	58.6	62.5	**70.2**
extra-large clamp	30.4	36.1	41.5	50.7	**69.0**

Table 7. Instance semantic segmentation results (mIoU(%)) of different methods on the YCB-Video dataset. Jointly training semantic segmentation module with keypoint offset module ($\mathcal{M}_{\mathcal{S},\mathcal{K}}$) obtains size information from the offset module and performs better, especially on large clamp and extra-large clamp. With the center voting module $\mathcal{M}_{\mathcal{C}}$ and the Mean-Shift clustering algorithm, further improvement of performance is obtained.

5. Conclusion

1. We propose a novel deep 3D keypoints voting network with instance semantic segmentation for 6DoF pose estimation, which outperforms all previous approaches in several datasets by large margins.

2. We also show that jointly training 3D keypoint with semantic segmentation can boost the performance of each other.

3. We believe the 3D keypoint based approach is a promising direction to explore for the 6DoF pose estimation problem

End