Iterative Optimisation with an Innovation CNN

Thesis Proposal Review

Gerard Kennedy

Supervisors: Robert Mahony, Xin Yu, Nick Barnes, Hongdong Li

Overview

- Motivation
- Innovation CNN Introduction
- Technical Talk: Initial Application
  - Object Pose Estimation
  - Formulation
  - Network Architecture
  - Evaluation
  - Design Choices
  - Initial Results
- Future work

Motivation

Variables In An Estimation Problem

- **Input:** the measured variables available to the estimation algorithm.
- **State:** the set of internal variables that summarises all the information in the system.
- **Output:** the variables that are required to be estimated.

State Estimation

- **State estimator:** an algorithm that enables the extraction of information about features of a system that are not explicitly provided by the data, via estimation of an underlying state representation.
- **System model:**

state

input data, measurements

$$\dot{\mathbf{X}}_t = f(\mathbf{X}_t, \mathbf{V}_t)$$
$$\mathbf{y}_t = h(\mathbf{X}_t)$$

desired output

map state to output

Online State Estimation

- Updating the desired information as new data becomes available
- Fundamental in robotics

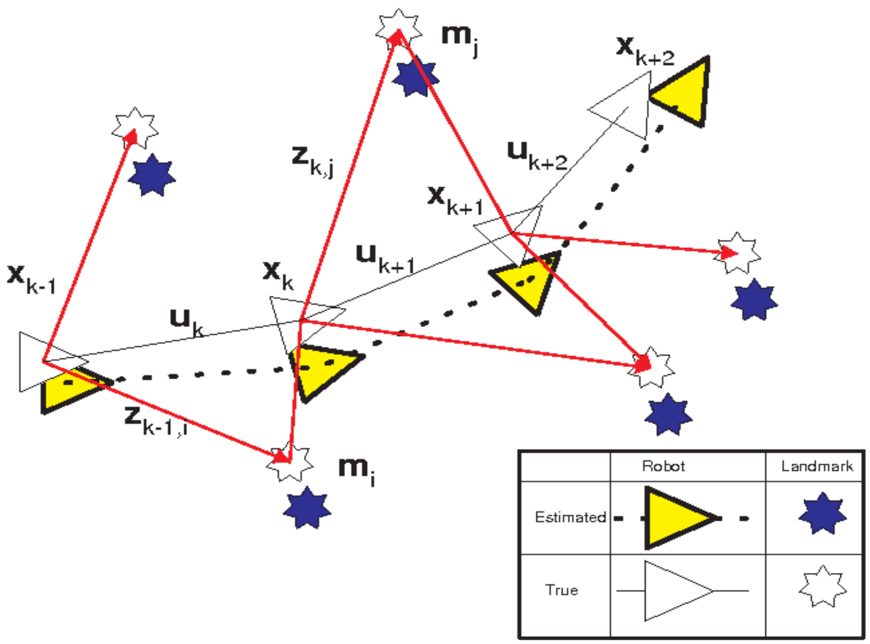


Image from: *Durrant-Whyte, Hugh and Tim Bailey. "Simultaneous Localisation and Mapping (SLAM) : Part I The Essential Algorithms." (2006).*

State Example: Visual Odometry

State model

Input: measurements

$$\dot{\hat{\mathbf{X}}}_t = f(\hat{\mathbf{X}}_t, \mathbf{V}_t) - \Delta$$

Output: landmark pose

$$\hat{\mathbf{y}}_t = h(\hat{\mathbf{X}}_t)$$

where

Innovation

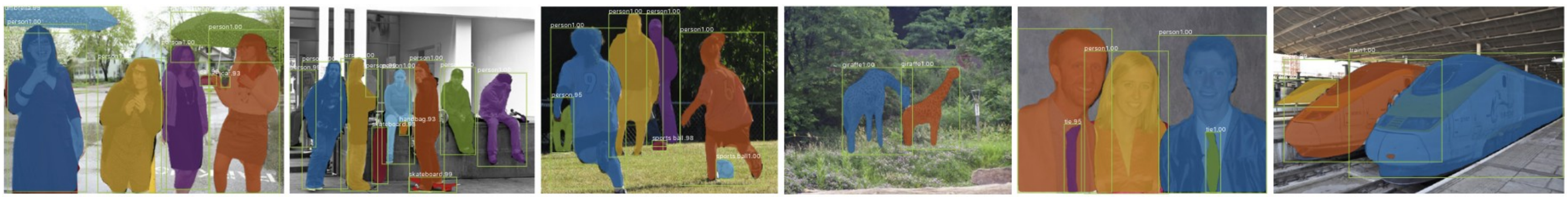
$$\Delta = K(\hat{\mathbf{y}}_t - \mathbf{y}_t)$$

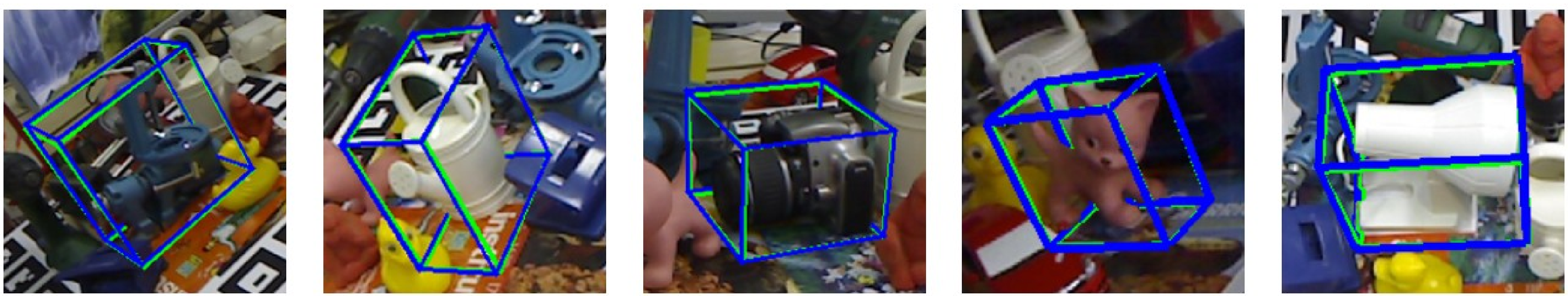
$$\mathbf{y} = [p_1, p_2, ..., p_N]$$

Measurement: landmark pose

Offline State Estimation

- Estimating output given fixed input data
- Fundamental Computer Vision Problem

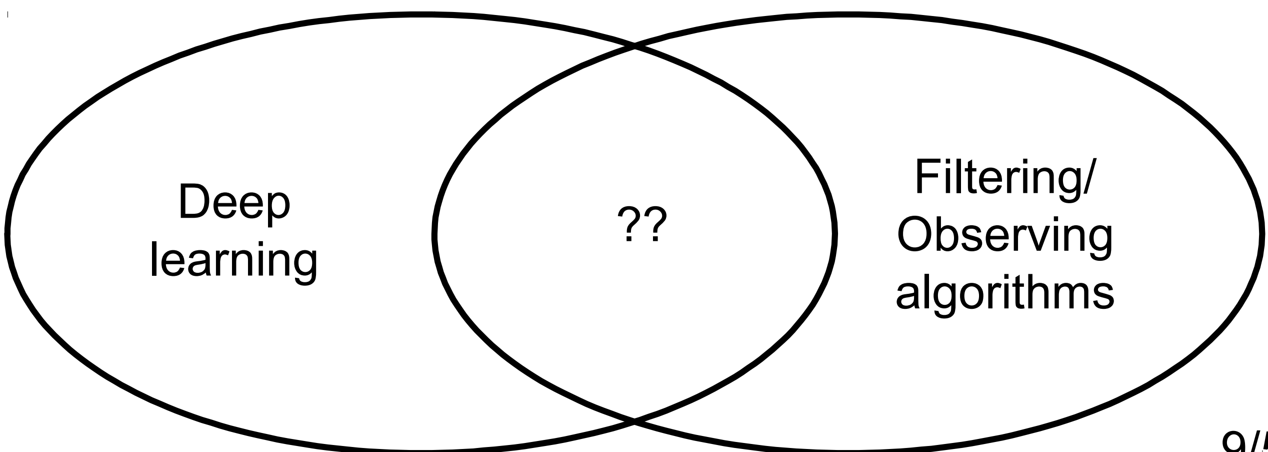




Top: *K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2017.*
Bottom: *M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," 2017.*

Problem Motivation

- Offline estimation = learning
- Online estimation = filtering/observing
- **How could these be combined and applied to both categories of estimation problem?**



Innovation CNN

The Innovation CNN

Innovation: the difference between the current state and the predicted state

Typical Approach

$$\hat{\mathbf{X}}_t = H^{-1}(\mathbf{I}_t)$$

$$\hat{\mathbf{y}}_t = h(\hat{\mathbf{X}}_t)$$

CNN

Proposed Approach

$$\dot{\hat{\mathbf{X}}}_t = f(\hat{\mathbf{X}}_t, \mathbf{V}_t) - \Delta$$

$$\hat{\mathbf{y}}_t = h(\hat{\mathbf{X}})$$

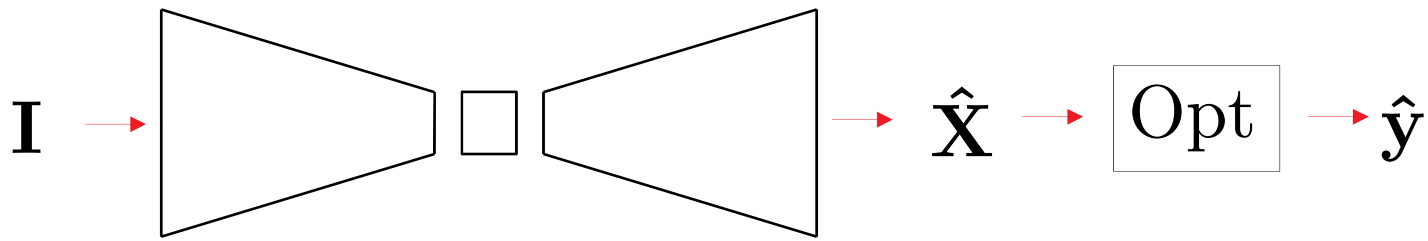
where

$$\Delta = K(\hat{\mathbf{y}}_t - \mathbf{y}_t)$$

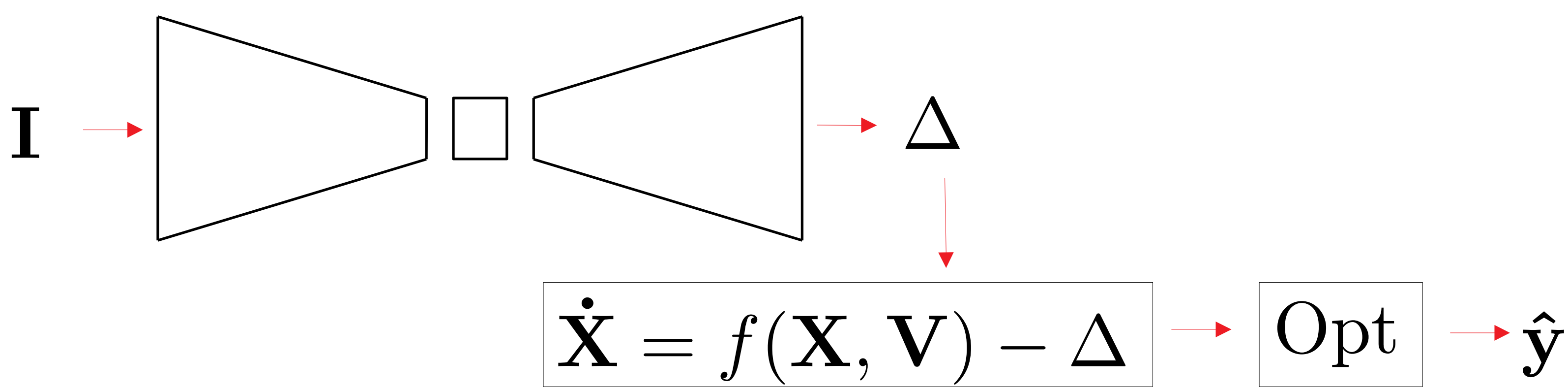
CNN

The Innovation CNN

- Classical:



- Innovation:



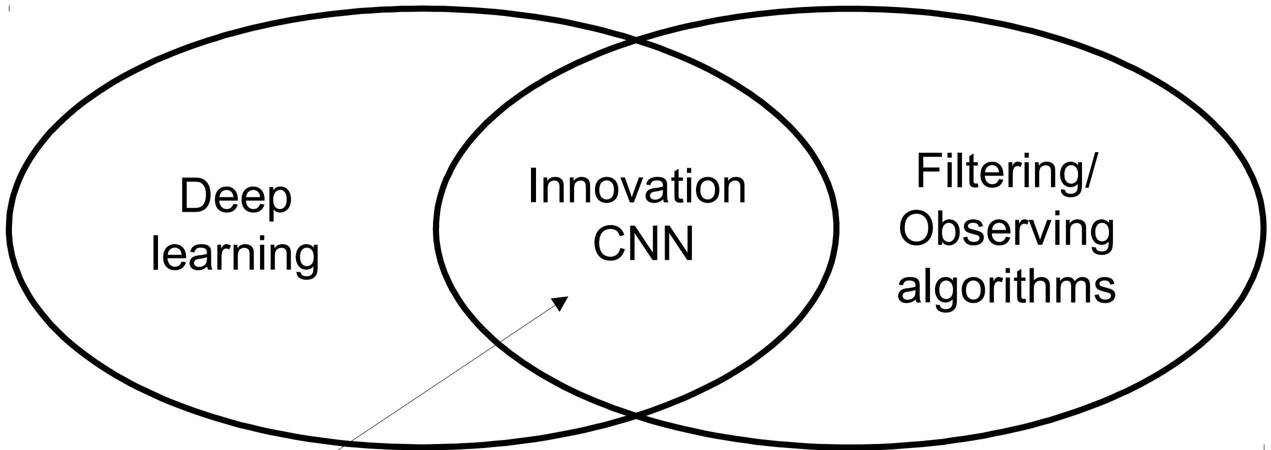
The Innovation CNN

- For **online** state estimation
  - Problems often `solved’ with filtering algorithm (eg. Kalman filter)
  - An Innovation CNN can be implemented to learn the innovation term
  - The measurements are provided by the robot’s sensors and the initial state is typically identity

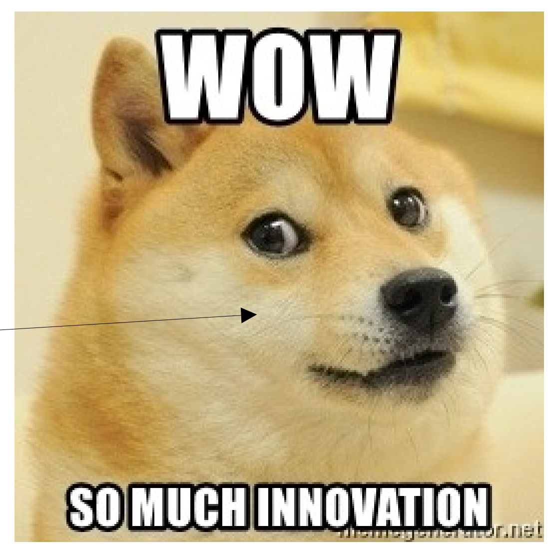
The Innovation CNN

- For **offline** state estimation
  - An Innovation CNN can be formulated from a CNN for offline state estimation by learning a suitable innovation term
  - The initial state estimate can be taken from the output of the original network
  - The initial estimate can be updated in an iterative framework

The Innovation CNN



Good idea



Obligatory tenuous meme

Good dog

Concept Demonstration

- Choose a trial problem: Object pose estimation (Offline)
- Select a pose estimation network
- Learn to estimate an innovation term which we can use to refine the state

Technical Talk

- **Initial application:** Object pose estimation from a single RGB image
- Offline estimation problem
- Typically implemented with CNN

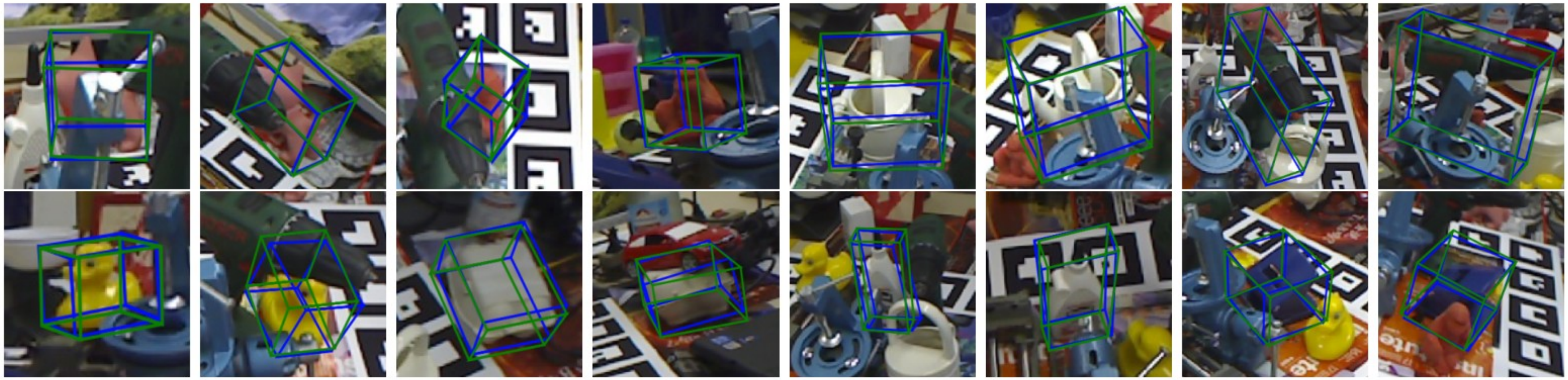


Image from: *S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in CVPR, 2019*

Object Pose Estimation

- 6D object pose estimation from a single RGB image

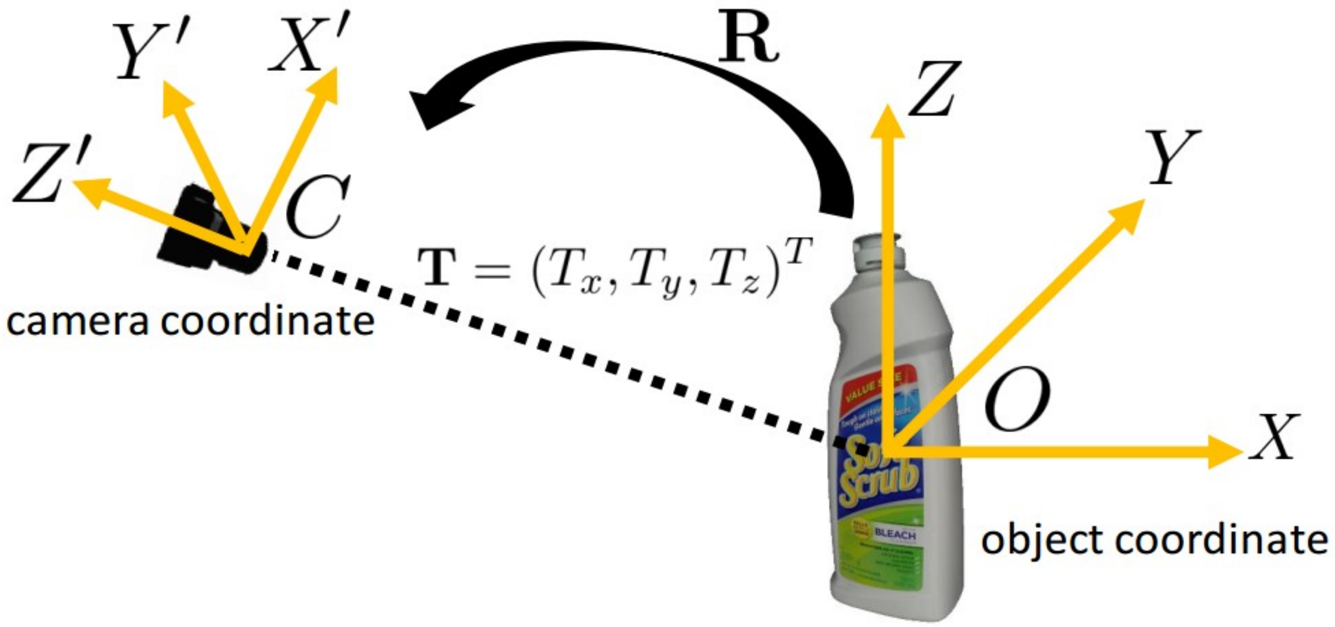


Image from: *Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," 2017*

Object Pose Estimation

- Highly challenging due to viewpoint ambiguity and object symmetries

Image from: *S. Hinterstoisser, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes."*

Applications

- Robotic manipulation and grasping (Amazon picking challenge)
- Scene understanding
- Virtual and augmented reality

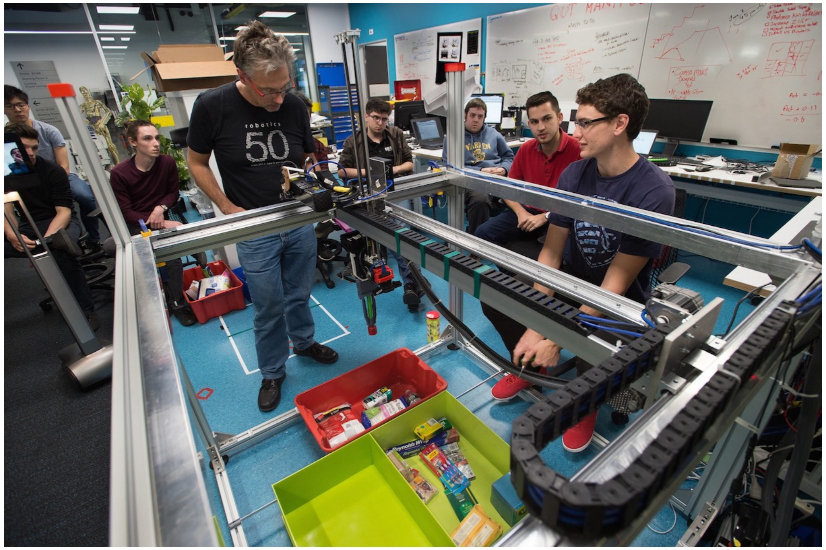


Photo: *Anthony Weate/QUT*

https://phys.org/news/2018-11-augmented-reality.html

Common Approaches

- End-to-end regression
- Classification via discretised pose space
- Regression to intermediate representation (keypoints) followed by PnP
- Pose refinement

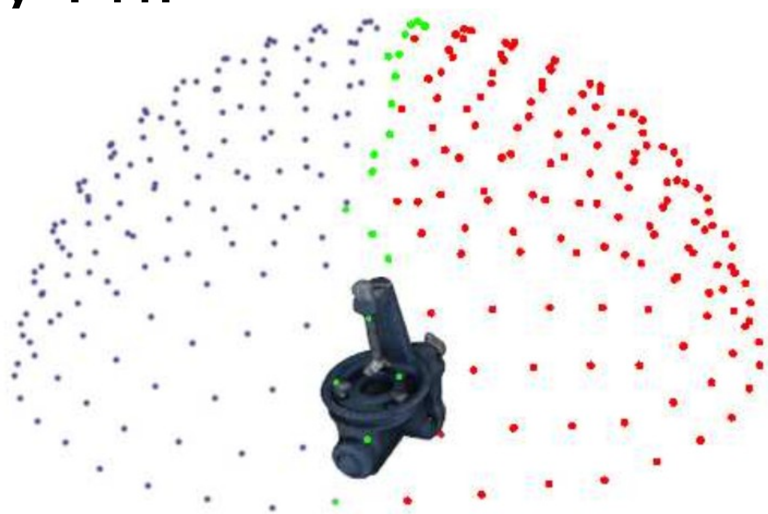
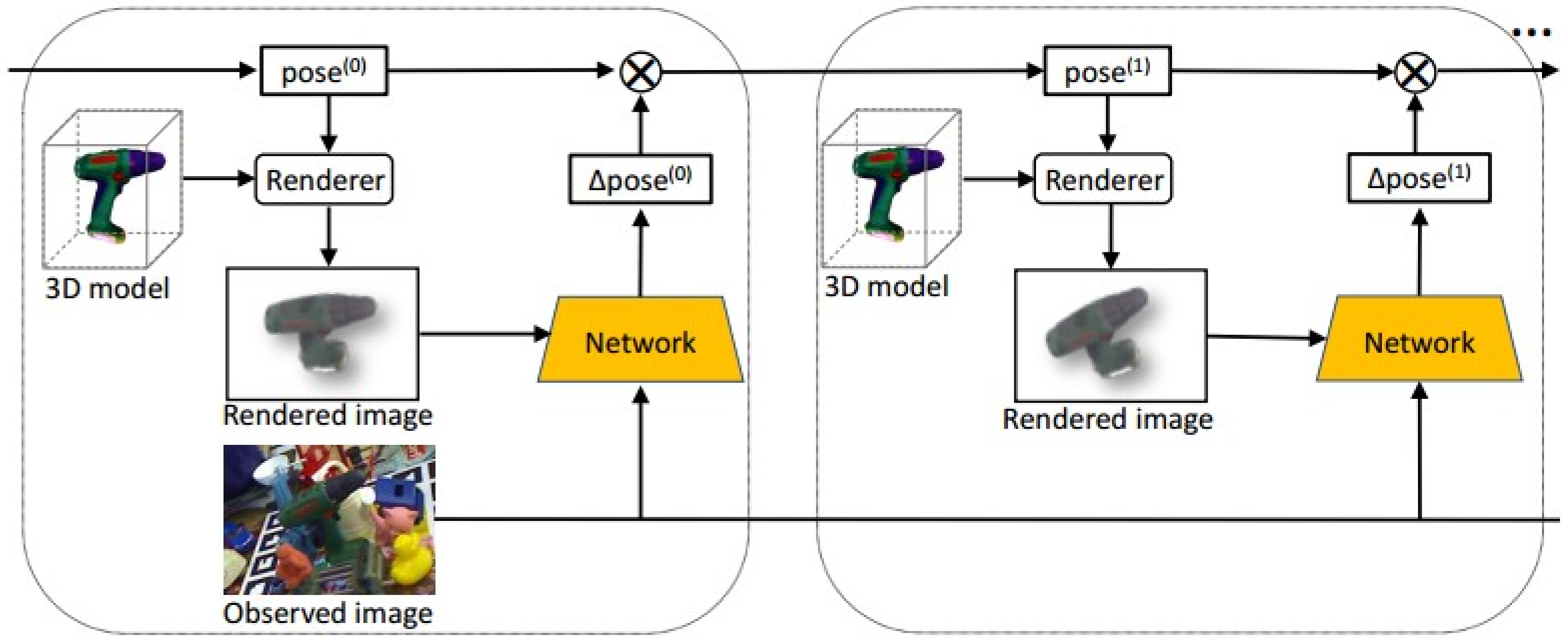


Image from: *W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," 2017*

Pose Refinement



$$L_{pose}(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{n}\sum_{i=1}^{n} \|(\mathbf{R}\mathbf{x}_i + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{x}_i + \mathbf{t})\|_1$$

Image from: *Y. Li, G. Wang, X. Ji,Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation, "International Journal of Computer Vision, vol. 128, no. 3, p. 657–678, Nov 2019. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01250-9*

PVNet

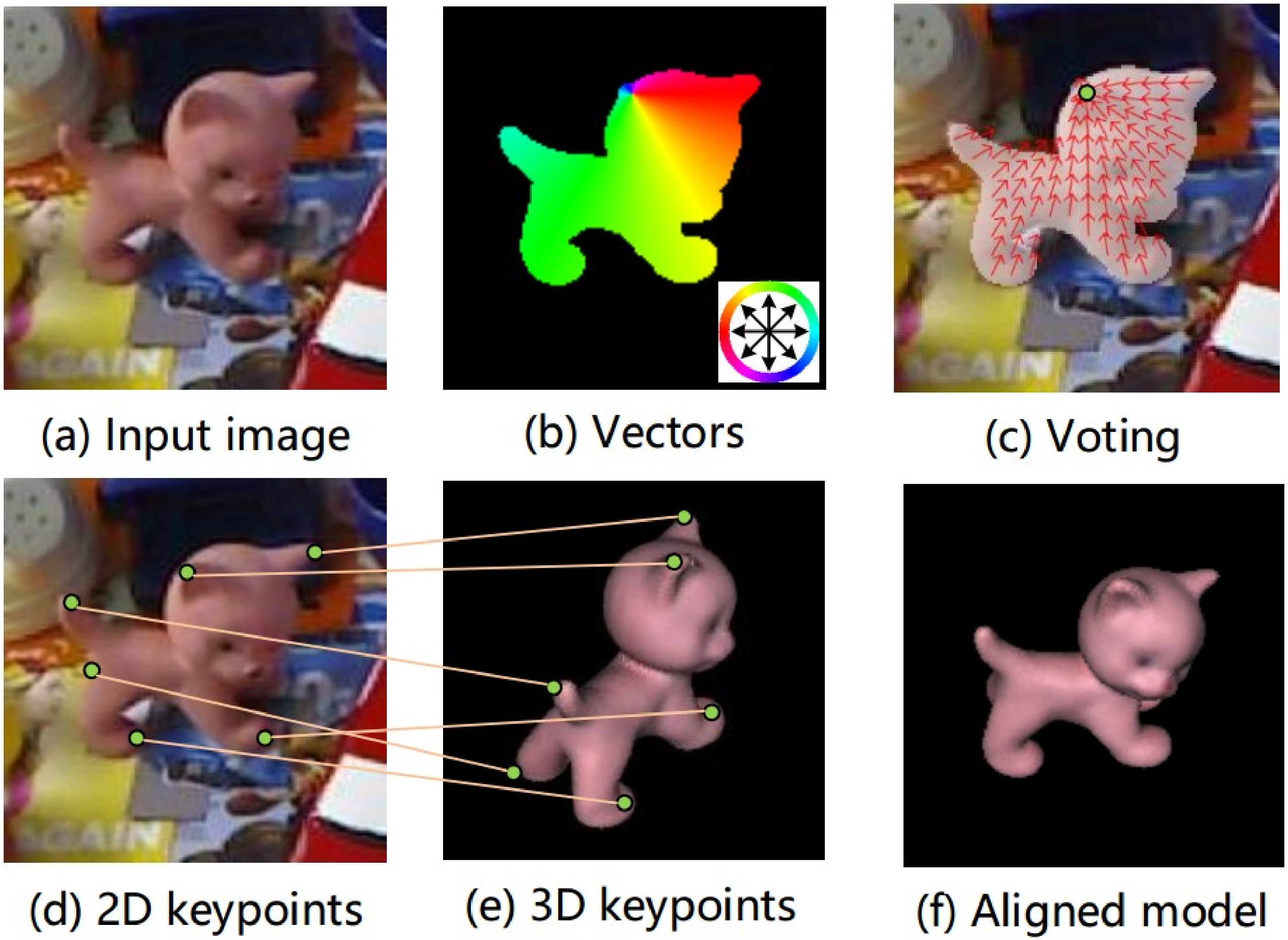


Image from: *S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in CVPR, 2019*

Problem Formulation

Innovation CNN for Pose Estimation

Model:

$$\mathbf{X}_{t+1} = \mathbf{X}_t$$

Then:

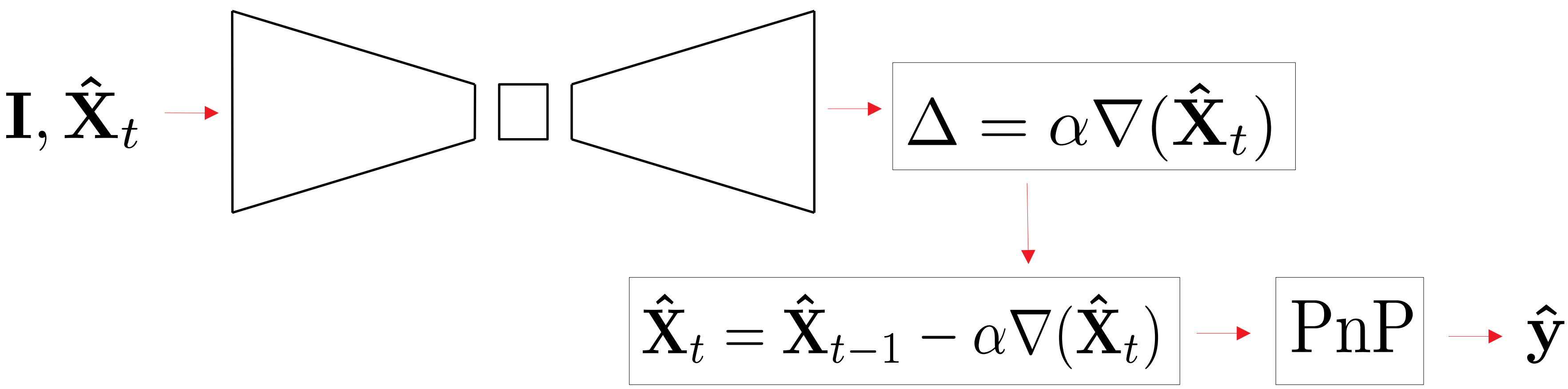
$$\hat{\mathbf{X}}_{t+1} = \hat{\mathbf{X}}_t - \Delta$$

This can be formulated as a S.G.D problem:

$$\text{let: } \Delta = \alpha \nabla(\hat{\mathbf{X}}_t)$$

$$\text{then: } \hat{\mathbf{X}}_{t+1} = \hat{\mathbf{X}}_t - \alpha \nabla(\hat{\mathbf{X}}_t)$$

Innovation CNN for Pose Estimation

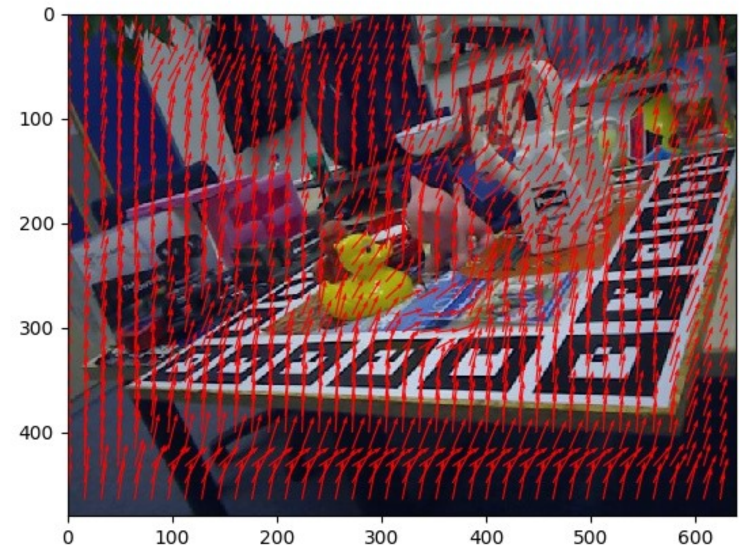


Problem Formulation

- State estimate:

$$\hat{\boldsymbol{\eta}}_{ij}^{k} = \hat{\boldsymbol{\xi}}^{k} - \hat{\boldsymbol{\xi}}_{ij}$$

$$\hat{\mathbf{X}}_{ij}^{k} = \frac{\hat{\boldsymbol{\eta}}_{ij}^{k}}{||\hat{\boldsymbol{\eta}}_{ij}^{k}||_2} \in \mathbb{R}^{2\times\mathcal{K}\times M\times N}$$



where $\hat{\boldsymbol{\xi}}^{k}$ is the pixel location of keypoint $k \in \mathcal{K}$,

$\hat{\boldsymbol{\xi}}_{ij}$ is pixel location i, j within an image with dimensions M, N,

$\hat{\boldsymbol{\eta}}_{ij}^{k}$ is the unit vector from pixel i, j to keypoint k,

and $\hat{\boldsymbol{X}}_{ij}^{k}$ is the state estimate.

Problem Formulation

- State gradient:

$$\Phi = \frac{1}{2}||\mathbf{X}^k - \hat{\mathbf{X}}_{ij}^k||_1^2$$

$$\therefore \nabla_{\hat{\mathbf{X}}_{ij}^k}\Phi = \frac{1}{2}\nabla_{\hat{\mathbf{X}}_{ij}^k}||\mathbf{X}^k - \hat{\mathbf{X}}_{ij}^k||_1^2$$

$$= -(\mathbf{X}^k - \hat{\mathbf{X}}_{ij}^k)$$

where $\mathbf{X}^k$ is the ground truth vector field,

$\nabla_{\hat{\mathbf{X}}_{ij}^k}$ is the gradient operator,

and $\nabla_{\hat{\mathbf{X}}_{ij}^k}\Phi$ is the state gradient.

Problem Formulation

- State update:

$$\hat{\mathbf{X}}_{ij}^{k}(t+1) = \hat{\mathbf{X}}_{ij}^{k}(t) - \alpha \widehat{\nabla_{\hat{\mathbf{X}}_{ij}^{k}} \Phi}(t)$$

where $\hat{\boldsymbol{X}}$ is the state estimate, t is the timestep/iteration, $\alpha \in (0,1)$ is the step size, and $\widehat{\nabla_{\hat{\mathbf{X}}_{ij}^{k}} \Phi}$ is the state gradient.

Note: the step size α is substituted with $\sigma \in (0,1)$ during training, and with $\delta \in (0,1)$ during evaluation.

Innovation CNN for Pose Estimation

Image

Initial state estimate

$$\mathcal{I} \quad \hat{\mathbf{X}}_{ij}^{k}(0)$$

State estimate (vector field)

$$\hat{\mathbf{X}}_{ij}^{k}(t)$$

Innovation CNN

$$\widehat{\nabla_{\hat{\mathbf{X}}_{ij}^{k}} \Phi}$$

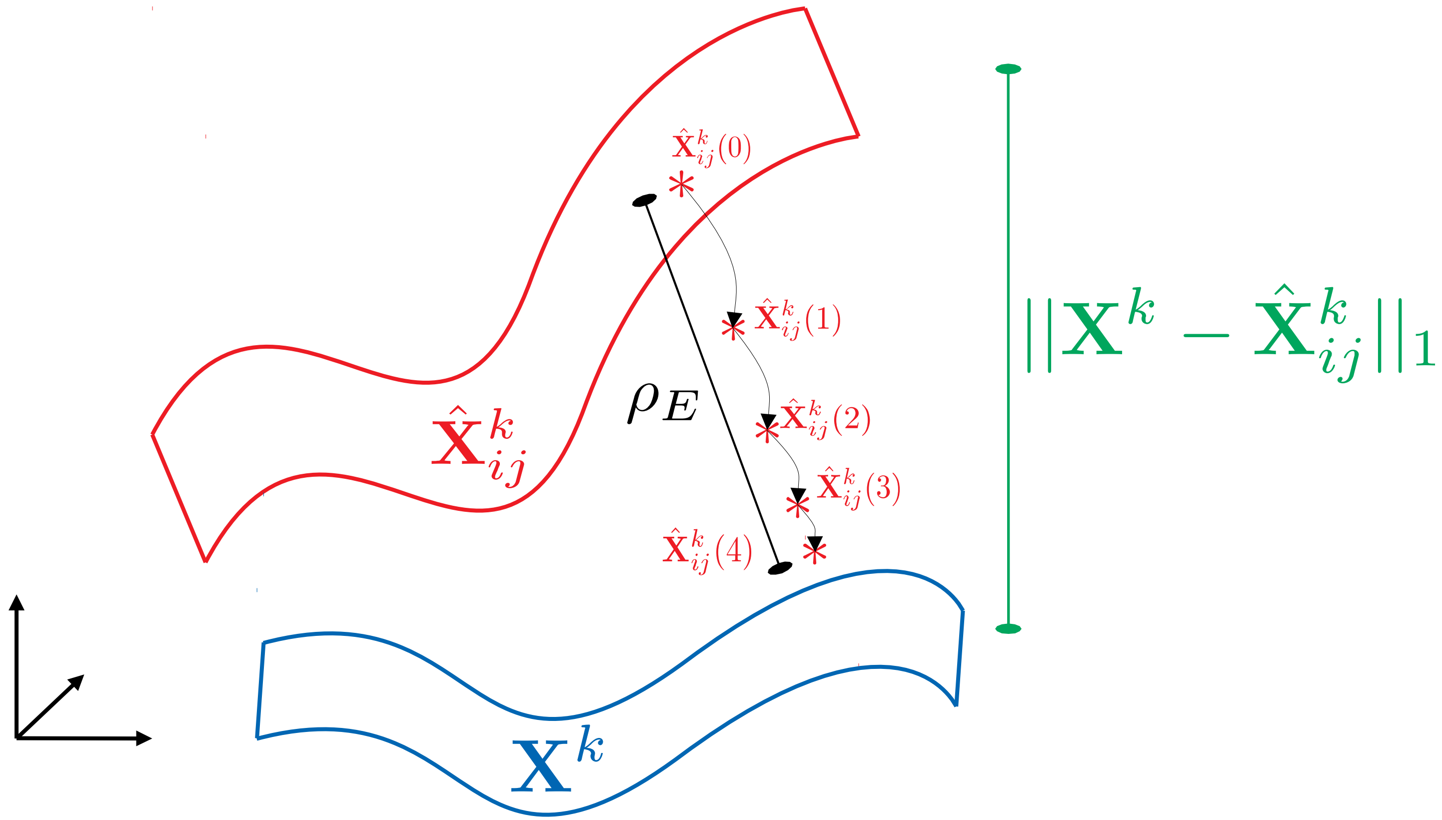
State gradient (gradient of vector field)

Update State estimate

$$\hat{\mathbf{X}}_{ij}^{k}(t+1) = \hat{\mathbf{X}}_{ij}^{k}(t) - \alpha \widehat{\nabla_{\hat{\mathbf{X}}_{ij}^{k}} \Phi}(t)$$

$$\hat{\mathbf{X}}_{ij}^{k}(T)$$

Iterative Refinement



$\rho_E = T_E \delta$ (the 'interpolation distance')

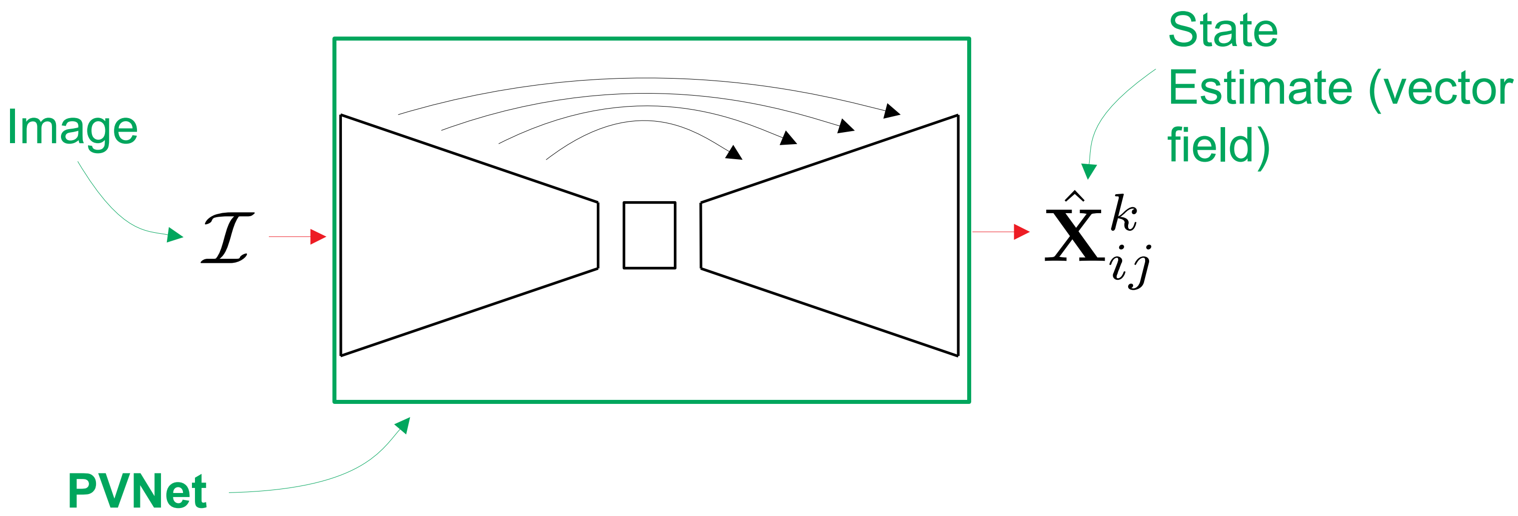
Innovation CNN for Pose Estimation

Algorithm 1 Iterative Optimisation with Innovation CNN

1: Choose $0 < \alpha < 1$
2: Choose $T > 0$ ▷ Maximum #iterations
3: $\hat{\boldsymbol{X}}_{ij}^{k}(0) \leftarrow$ PVNet
4: **for** $t = 1 \rightarrow T$ **do**
5: $\quad \widehat{\nabla_{\Phi}} \leftarrow$ Innovation CNN
6: $\quad \hat{\boldsymbol{X}}_{ij}^{k}(t) = \hat{\boldsymbol{X}}_{ij}^{k}(t-1) + \alpha \widehat{\nabla_{\Phi}}(t)$
7: pose = PnP($\hat{\boldsymbol{X}}_{ij}^{k}(T)$)

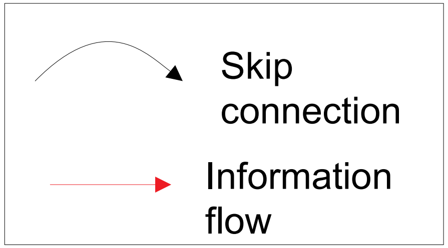
Network Architecture

Network Architecture

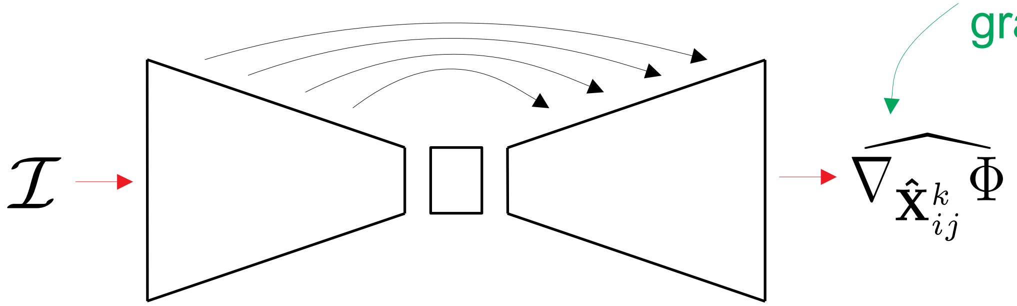


PVNet Loss

$$\Phi = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} ||\boldsymbol{X}^{k} - \hat{\boldsymbol{X}}_{ij}^{k}||_{1}^{2}$$

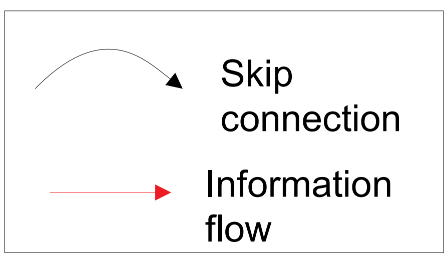


Network Architecture



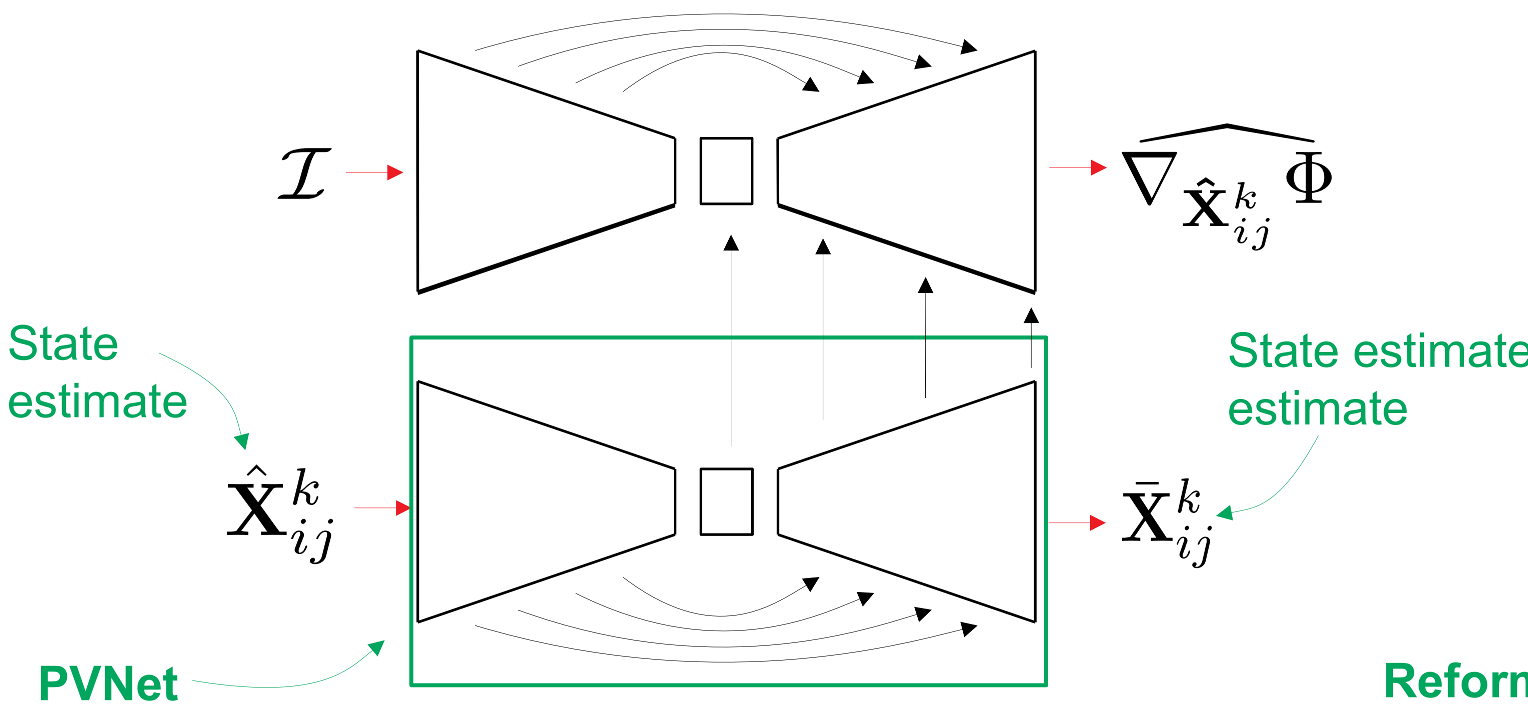
State gradient

Reformulated loss

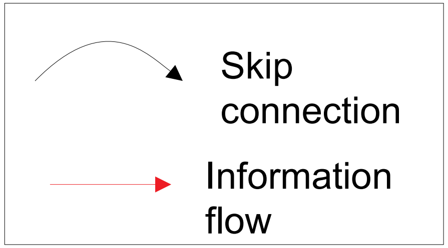


$$\mathcal{L}_{(\nabla_{\hat{\mathbf{x}}_{ij}^k}\Phi)} = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} ||\widehat{\nabla_{\hat{\mathbf{x}}_{ij}^k}\Phi} - (\boldsymbol{X}^k - \hat{\boldsymbol{X}}_{ij}^k)||_1$$

Network Architecture

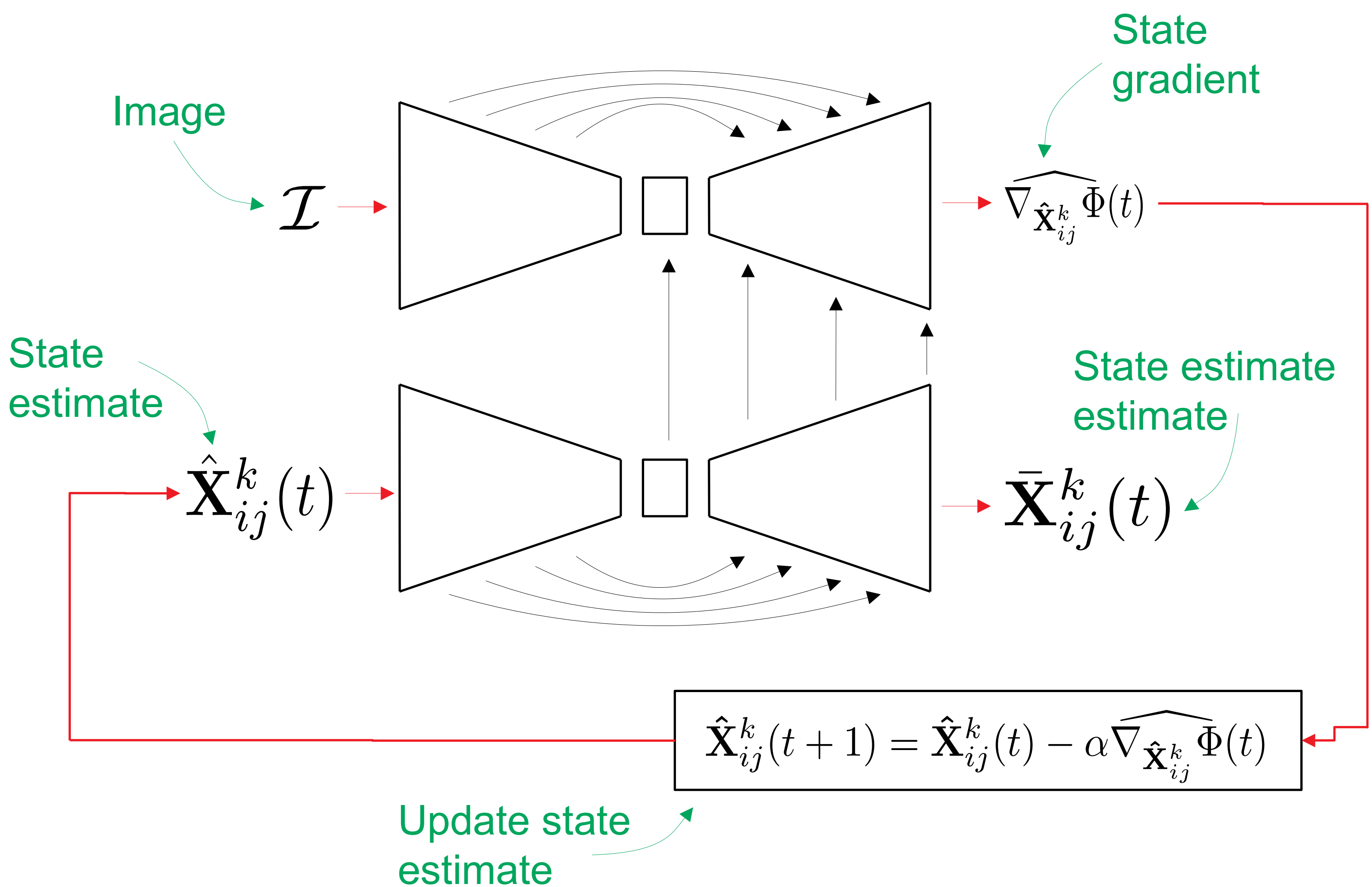


Reformulated loss

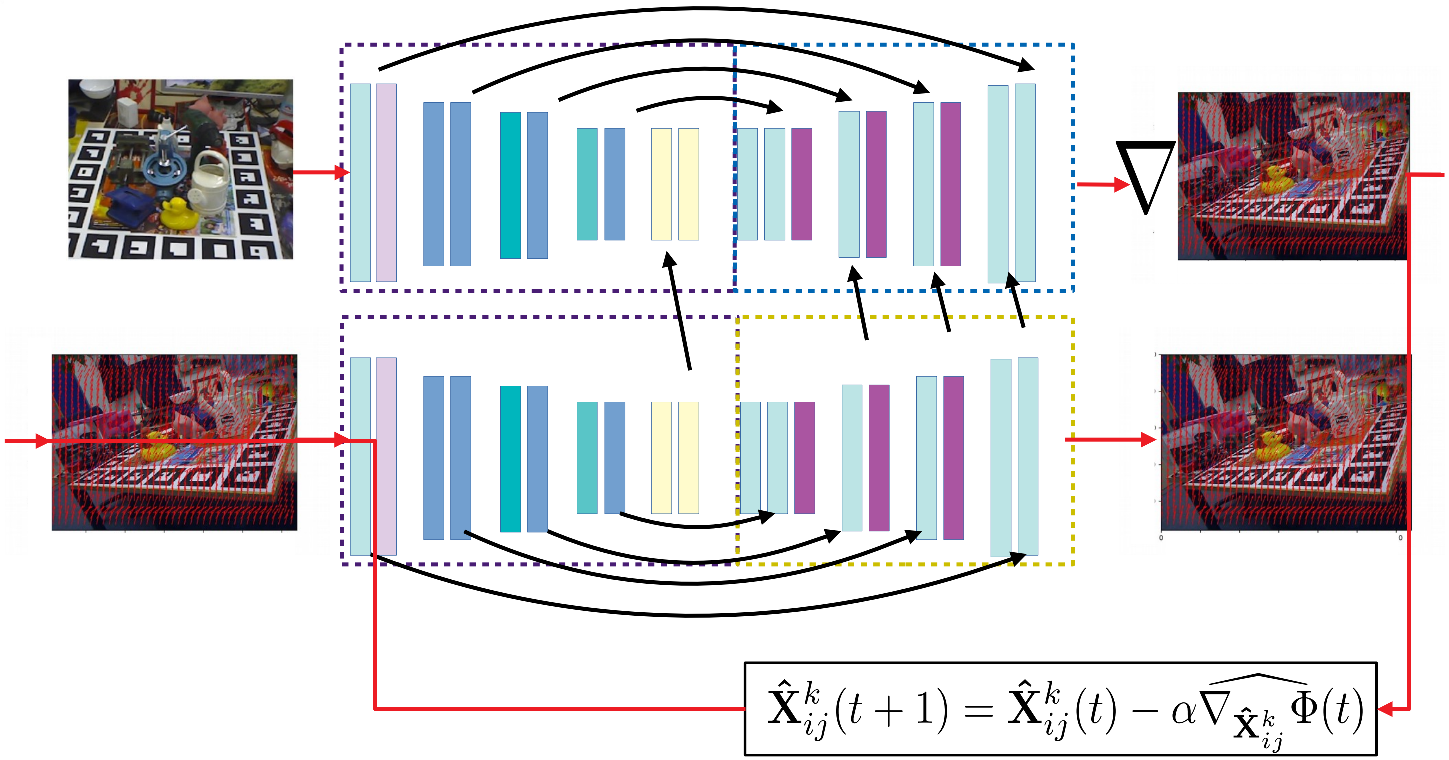


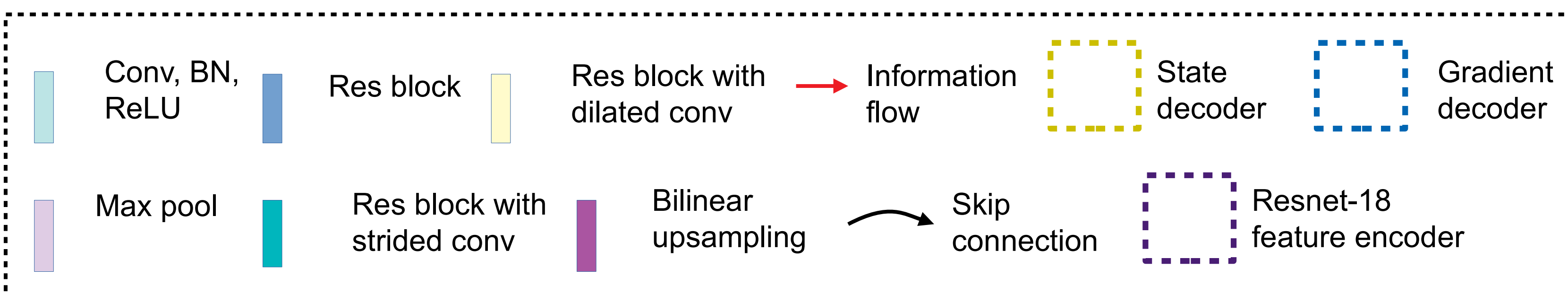
$$\mathcal{L}_{\boldsymbol{X}} = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} ||\hat{\boldsymbol{X}}_{ij}^{k} - \bar{\boldsymbol{X}}_{ij}^{k}||_1$$

Network Architecture



Network Architecture





Network Training

$$\mathcal{L}(t) = \mathcal{L}_{(\nabla_{\hat{\mathbf{x}}_{ij}^k}\Phi)}(t) + \gamma\mathcal{L}_{\boldsymbol{X}}(t)$$

$$\mathcal{L}_{\boldsymbol{X}}(t) = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} ||\hat{\boldsymbol{X}}_{ij}^k(t) - \bar{\boldsymbol{X}}_{ij}^k(t)||_1 \text{ , and}$$

$$\mathcal{L}_{(\nabla_{\hat{\mathbf{x}}_{ij}^k}\Phi)}(t) = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} ||\widehat{\nabla_{\hat{\mathbf{x}}_{ij}^k}\Phi}(t) - (\boldsymbol{X}^k - \hat{\boldsymbol{X}}_{ij}^k(t))||_1$$

Evaluation

Evaluation Metrics

- Standard metrics:

$$\mathrm{ADD} = \frac{1}{m} \sum_{\mathbf{x} \in \mathcal{M}} ||(\mathbf{R}\mathbf{x} + \mathbf{T}) - (\hat{\mathbf{R}}\mathbf{x} + \hat{\mathbf{T}})||_2$$

where $\mathcal{M}$ denotes the set of 3D model points, and m is the number of points.

- The pose is considered correct if the average distance is <10% of the 3D model diameter
- The metric reported is the % of correct poses

Evaluation Metrics

- Standard metrics:

$$2\text{d Proj} = \frac{1}{|V|} \sum_{\mathbf{v} \in V} ||\mathbf{P}\mathbf{X}_{ij}^{k}\mathbf{v} - \mathbf{P}\hat{\mathbf{X}}_{ij}^{k}\mathbf{v}||_2$$

where V is the set of all object model vertices, and $\mathbf{P}$ is the camera matrix.

- The pose is considered correct if the average error is <5 pixels
- The metric reported is the % of correct poses

Evaluation Metrics

- Problem specific metric:

$$\mathrm{Norm}(\hat{\mathbf{X}}_{ij}^{k} - \mathbf{X}_{ij}^{k}) = \frac{1}{N}\frac{1}{||\mathcal{S}||}\sum_{n=0}^{N}\sum_{(i,j)\in\mathcal{S}} ||\hat{\mathbf{X}}_{ij,n}^{k} - \mathbf{X}_{ij,n}^{k}||_2^2$$

where N is the number of samples,
and $\mathcal{S}$ is the segmentation mask.

Design Choices

Choosing Parameters

- Baseline PVNet: 47.2% (ADD)

σ	ρ_T	δ	ρ_E	mean(ADD)	mean(% increase ADD)	mean(% decrease norm(X-X^))	mean(ADD) T/P values
1	1	1	2	0.424 +/- 0.04	-9.81 +/- 8.47	9.29 +/- 1.37	-2.12/0.05
1	2	1	4	0.520 +/- 0.03	10.33 +/- 6.70	6.75 +/- 1.33	3.76/0.001
0.9	1.8	0.9	3.6	0.518 +/- 0.03	9.48 +/- 7.17	7.56 +/- 1.02	3.61/0.001
0.6	1.2	0.6	2.4	**0.539 +/- 0.03**	**14.50 +/- 6.56**	**17.99 +/- 0.89**	**5.26/0.001**
0.6	2.4	0.6	4.8	0.464 +/- 0.01	-1.83 +/- 3.66	2.11 +/- 0.34	-5.16/0.001
0.3	1.2	0.3	2.4	0.471 +/- 0.01	0.44 +/- 2.79	1.91 +/- 0.34	-0.65/0.2+

Table 3.1: Study of Iteration Parameters. All mean and standard deviation values were computed from the last 20 epochs of training, from a total of 50 epochs.

$$\rho_T = \sigma T_T$$

$$\rho_E = \delta T_E$$

Choosing Parameters

GE/+SA	$\nabla_\Phi / \widehat{\nabla_\Phi}$	L_{∇_Φ}	initial est	mean(ADD)	mean(% increase ADD)	mean(% decrease norm(X-X^))	mean(ADD) T/P values	GE/+SA	$\nabla_\Phi / \widehat{\nabla_\Phi}$	L_{∇_Φ}	initial est
GE+SA	∇_Φ	**scaled**	**PVNet**	0.511 +/- 0.036	8.29 +/- 7.82	12.91 +/- 2.55	2.13/0.05				
GE+SA	$\widehat{\nabla_\Phi}$	**scaled**	**PVNet**	0.508 +/- 0.043	7.51 +/- 9.12	12.50 +/- 3.25	1.38/0.2				
GE+SA	$\widehat{\nabla_\Phi}$	**unscaled**	**PVNet**	0.470 +/- 0.066	-0.16 +/- 13.67	13.60 +/- 2.55	0.01/0.5+				
GE	∇_Φ	**unscaled**	**PVNet**	0.506 +/- 0.034	7.40 +/- 8.06	14.78 +/- 1.07	2.08/0.05				
GE+SA	∇_Φ	**unscaled**	**PVNet**	0.539 +/- 0.030	14.50 +/- 6.56	17.99 +/- 0.89	4.79/0.001				
GE+SA	∇_Φ	**unscaled**	**GT +/- 10%**	0.460 +/- 0.008	-2.81 +/- 2.61	2.04 +/- 0.20	-10.85/0.001				
GE+SA	∇_Φ	**unscaled**	**GT +/- 1%**	0.460 +/- 0.007	-2.74 +/- 2.07	1.84 +/- 0.11	-13.83/0.001				

Table 3.2: Training Parameters. Experiments are colour-coded based on which design choices are being compared. The results of the experiment that performed best for a given pair of design choices is highlighted with the corresponding colour. T-value is obtained from Welch's T-test of the mean(ADD) compared to the original PVNet distribution: 0.472+/-0.0067. P-value is obtained from corresponding probability that the two means come from separate distributions.

Choosing Parameters

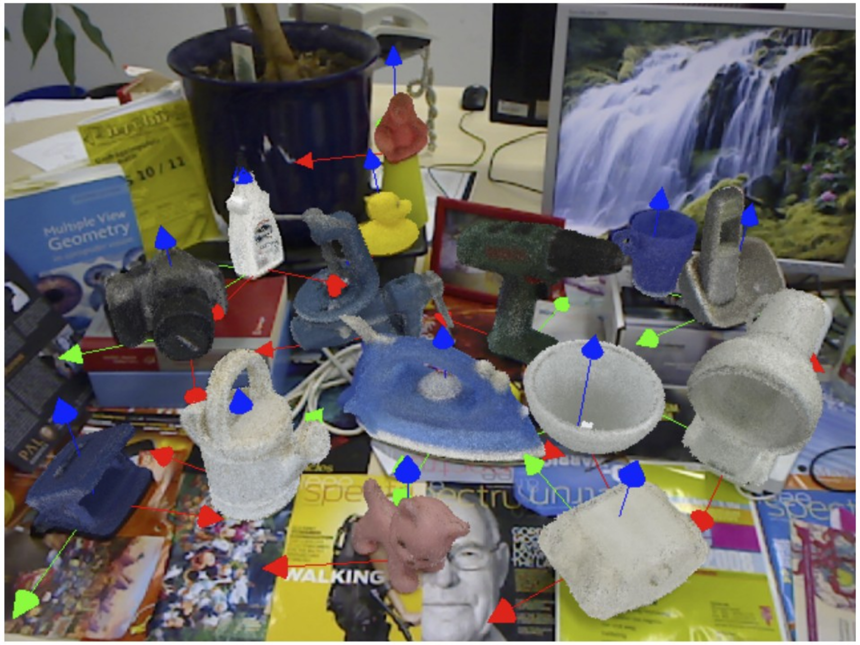
Loss	$\sigma, \rho_T, \delta, \rho_E$	mean(ADD)	mean(% increase ADD)	mean(% decrease norm(X-X^))	mean(ADD) T/P values
IRR	1,2,1,4	0.401 +/- 0.04	-15.04 +/- 8.51	4.79 +/- 1.14	3.14/0.01
	1,4,1,4	0.457 +/- 0.03	-3.12 +/- 6.83	1.41 +/- 1.01	
Innovation CNN	1,2,1,4	**0.520 +/- 0.03**	**10.33 +/- 6.70**	**6.75 +/- 1.33**	**3.76/0.001**
	1,4,1,4	0.406 +/- 0.03	-13.95 +/- 7.72	3.21 +/- 1.46	

Tal : IRR: Backprop after all iterations. Innovation CNN: Backprop each iteration. Both experiments use: $\sigma = 1$, $\rho_T = 2$, $\delta = 1$, $\rho_E = 4$. T-value is obtained from Welch's T-test of the mean(ADD) compared to the original PVNet distribution: 0.472+/-0.0067. P-value is obtained from corresponding probability that the two means come from separate distributions.

Initial Results

Experiments

- All experiments were undertaken on the Linemod dataset [1]
- Qualitative results are shown for Linemod's `Ape' object



[1]

[1] *S. Hinterstoisser, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes."*

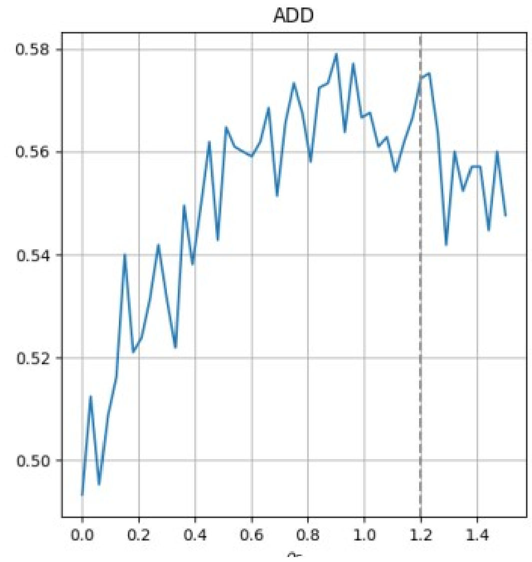
Qualitative Results

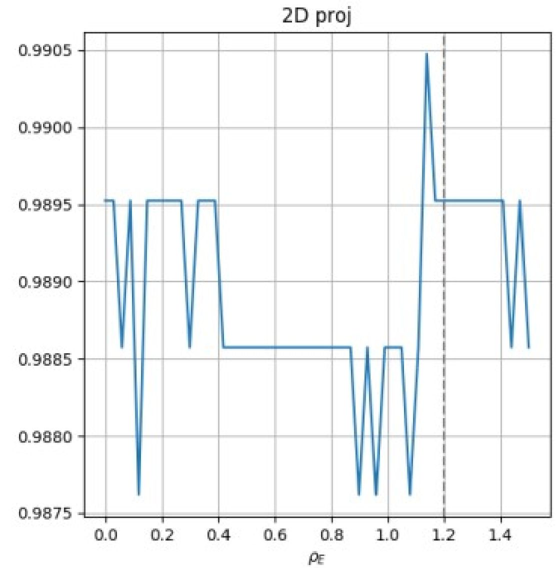
- Network trained with:

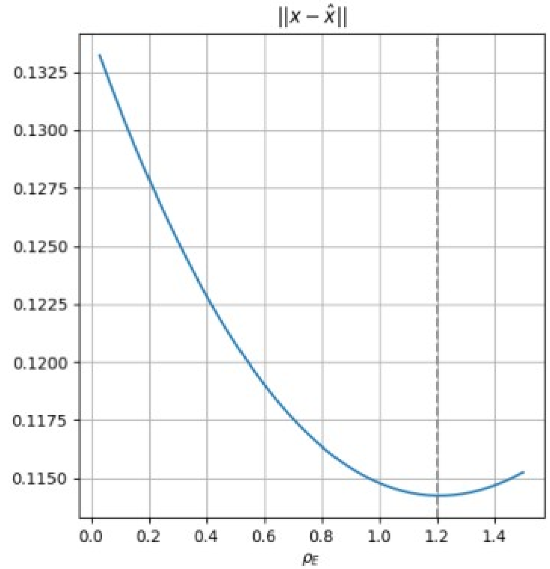
$$\sigma = 0.3$$
$$T_T = 4$$
$$\rho_T = T_T \sigma = 1.2$$

- Network evaluated with:

$$\delta = 0.03$$
$$T_E = 50$$
$$\rho_E = T_E \delta = 1.5$$







Quantitative Results

- Initial PVNet estimate: 47.2% (ADD metric)
- After iterative refinement: **59.8%**
- Performance increase: **~27%** (ADD)
- Decrease of **~18%** ($\mathrm{Norm}(\hat{\mathbf{X}}_{ij}^{k} - \mathbf{X}_{ij}^{k})$)

Conclusions

- We reformulated PVNet to an Innovation CNN for object pose estimation
- Obtained an increase in performance of ~27% on the Ape dataset, using the ADD metric

Next Steps

- Train and evaluate on remaining object categories of Linemod dataset
- Hope we get a similar performance increase

TPR: Future Work Proposal

Pose Estimation

- Test on remaining Linemod objects ASAP
- Also test on other standard datasets for object pose estimation (Linemod Occlusion and YCP)
- Incorporate into a more sophisticated pose estimation pipeline

Depth Estimation

- Try the same idea on depth estimation from an RGB image

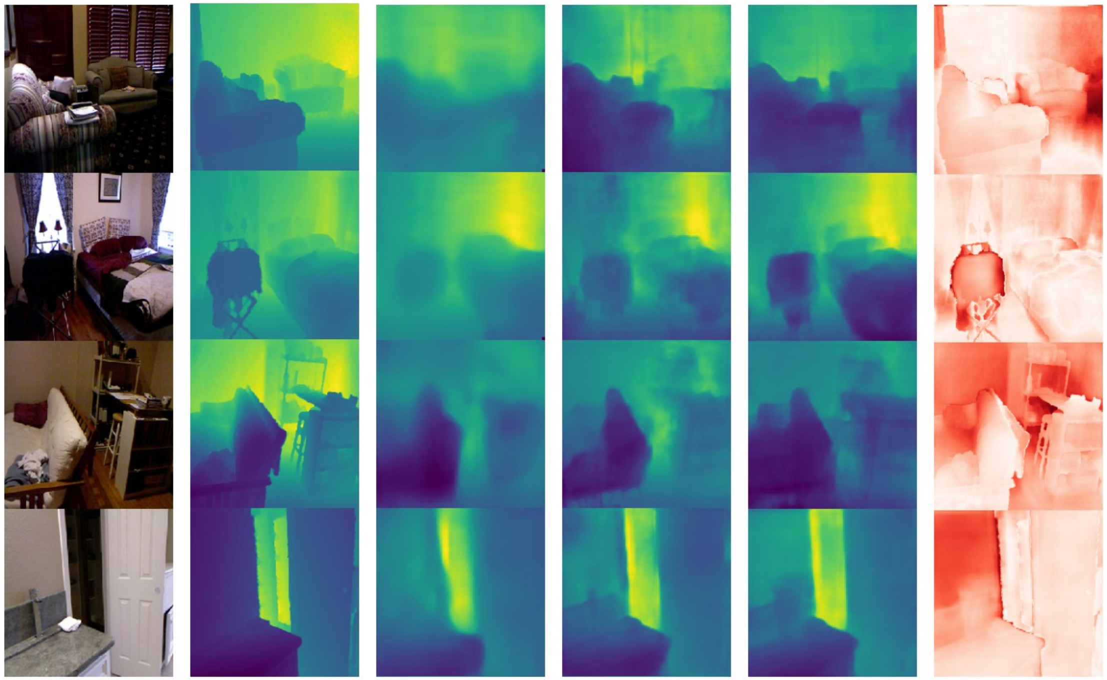


Image from: *D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "Fastdepth: Fast monocular depth estimation on embedded systems,"* 2019.

Online Estimation

- In our example offline problem the system state is modeled as being stationary. But we **could** have any system.
- Eg. visual odometry observer

AUSTRALIAN CENTRE FOR
ROBOTIC
VISION

Australian Government
Australian Research Council

QUT

THE UNIVERSITY of ADELAIDE

THE AUSTRALIAN NATIONAL UNIVERSITY

MONASH University

CSIRO

UNIVERSITY OF OXFORD

Inria
INVENTORS FOR THE DIGITAL WORLD

Imperial College London

ETH zürich

Georgia Tech