Learning Innovations for State Estimation

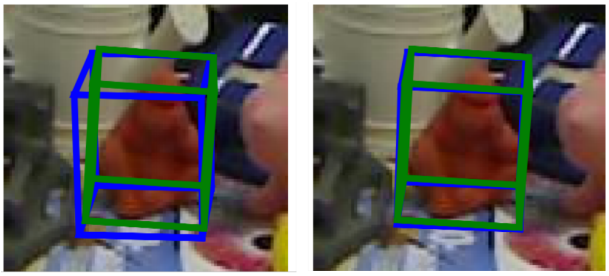
Gerard Kennedy[1], Jin Gao[1], Zheyu Zhuang[1], Xin Yu[2], and Robert Mahony[1]

***Abstract*— Deep learning algorithms such as Convolutional Neural Networks (CNNs) are currently used to solve a range of robotics and computer vision problems. These networks typically estimate the desired representation in a single forward pass and must therefore learn to converge from a wide range of initial conditions to a precise result. This is challenging, and has led to increased interest in the development of separate refinement modules which learn to improve a given initial estimate, thus reducing the required search space. Such modules are usually developed ad-hoc for each given application. In this work we propose a generic innovation-based CNN. Our CNN is implemented along with a stochastic gradient descent (SGD) algorithm to iteratively refine a given initial estimate. The proposed approach provides a general framework for the development of refinement modules applicable to a wide range of robotics problems. We apply this framework to object pose estimation and depth estimation and demonstrate significant improvement over the initial estimates, in the range of 4.2 - 8.1%, for both applications.**

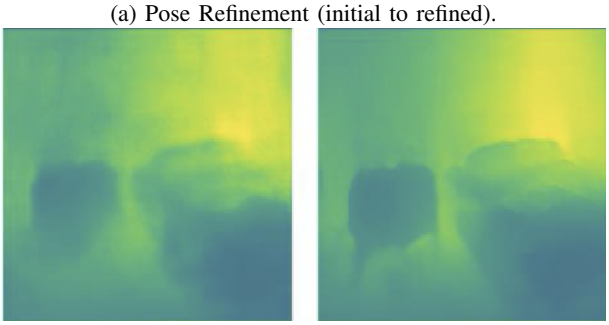
I. INTRODUCTION

Using a CNN to solve an image-based computer vision or robotics problem has become common in recent decades. Such problems often involve estimating a desired representation from a single RGB image. This approach is attractive as it allows use of a single monocular camera, rather than multiple cameras, stereo cameras, or other active sensors, all of which add weight and complexity to the vision system [37], [11]. However, estimation from a single image is challenging, and solutions often require additional refinement to obtain the desired accuracy. Two single image-based robotic vision problems for which ad-hoc refinement networks have been developed are object pose estimation [27], [41] and depth estimation [7].

Object pose estimation involves estimating the 6D pose (translation and orientation) of a specified object relative to the camera pose. This has a range of applications including robotic manipulation/grasping [5] and autonomous driving [4]. Recent state-of-the-art RGB pose estimation algorithms have regularly incorporated a separate refinement module [20], [41], [27]. Likewise, refining an initial depth estimate is also a popular problem, particularly when the initial estimate is obtained from a single image [18], [13], [33]. Depth estimation involves estimating a 1D per-pixel depth map, and is a key requirement of applications including autonomous driving [6], and agricultural robotics [22].



(a) Pose Refinement (initial to refined).



(b) Depth Refinement (initial to refined).

Fig. 1: Applying the Innovation CNN to object pose estimation and depth estimation. Left sub-figures show the initial estimate, while right sub-figures show our result.

For both these problems the refinement network is often developed as a separate network with an ad-hoc architecture developed specifically for the given problem. Furthermore, it is common for the refinement module to be applied multiple times to improve the state estimate [27], [7].

In this paper, we draw from the established principles of Stochastic Gradient Descent (SGD) and formulate a general framework that can be applied to a range of robotics and computer vision problems. We implement an iterative refinement framework based on formal principles, thus easing the requirement for the neural network to accurately estimate the desired output in a single forward pass. Specifically, we take a known network modify it to estimate a relative error, rather than estimating the state directly. The output of the principle network is used to drive an SGD update for the state estimate. Motivated by filtering theory [19], [1] we label the relative error the *innovation* and call our approach the *Innovation CNN*. By formally recognising the role of the state estimate and the SGD update term, we are able to employ existing algorithms and insights on step size choice and convergence analysis that have been developed for SGD (eg. using Wolfe

*This work was supported in part by the Australian Government Research Training Program Scholarship and in part by the Australian Research Council through the "Australian Centre of Excellence for Robotic Vision" CE140100016.

[1]Gerard Kennedy, Jin Gao, Zheyu Zhuang, and Robert Mahony are with the Research School of Engineering, Australian National University, Canberra, Australia firstname.lastname@anu.edu.au

[2]Xin Yu is with the University of Technology, Sydney, Australia xin.yu@uts.edu.au

conditions to control the step size) [9].

A key challenge in the design of refinement networks is the re-injection of the present estimate of state into the refinement module. For example, in pose refinement the present pose estimate is used to render a synthetic image that is then fed in parallel with the original image into the network [27], [41]. Our network architecture overcomes this difficulty by including an additional autoencoder network implemented in parallel with the principle network architecture, to encode the state estimate and inject this information using skip connections. This approach is general and can be applied to a wide range of existing networks for varying applications. In this work we apply our framework to two common robotic vision problems; object pose and depth estimation.

We evaluate our object pose refinement on two widely used benchmarks, LINEMOD [16] and Occlusion LINEMOD [3]. Our approach leads to an average improvement of 4.20% on LINEMOD and 8.12% on Occlusion LINEMOD relative to the initial estimate, in terms of the widely-used ADD(-S) metric. We further demonstrate that our method is particularly effective at refining relatively poor initial estimates. Specifically, we show relatively large improvement on the more difficult Occlusion LINEMOD dataset, along with particularly challenging objects of the LINEMOD dataset, such as ape (21.14% improvement) and duck (10.73% improvement) where the objects have poor surface texture.

We evaluate our depth refinement on the NYU Depth V2 dataset [30]. Our approach obtains an average improvement of 4.58% and 4.86% relative to the initial estimate, in terms of the RMSE and AME metrics respectively. We show that our method provides the greatest benefit to image regions associated with edges, boundaries, and finer details. An illustration of the refinement results achieved with the Innovation CNN is provided in Fig. 1.

II. RELATED WORK

A. Pose Estimation & Refinement

Research related to single-image object pose estimation can be sub-divided into four general areas: end-to-end regression, pose classification, regression to an intermediate representation, and pose refinement.

End-to-end pose regression from a single image is a highly challenging task that is approached in PoseNet [21] by separating and carefully balancing translation and rotation terms in the loss function, although this is for camera pose estimation, a slightly different problem. Similarly, [39] decouple or 'disentangle' translation and rotation terms by moving the centre of rotation to the centre of the object and representing translation in 2D image space. Recently, [15] achieved state-of-the-art via simultaneous regression to object class, bounding box, rotation, and translation.

Pose classification typically involves discretising $\mathrm{SO}(3)$, while the translation component is obtained via regression. Kehl *et al.* [20] approach this problem by decomposing 6D pose into viewpoint and discretised in-plane rotation.

A recent popular approach is to first regress to an intermediate representation such as keypoints, from which pose can be obtained via 2D-3D correspondences and a PnP algorithm [36], [32], [31], [40], [41], [28], [17]. Of these approaches, some, including [36], [32] regress to a set of bounding box corners, while others regress to vector fields [31], [34], [40], or dense correspondences.

Of the networks discussed above, several have additional refinement steps that can be added to improve results. For example, DeepIM [27] is presented as a refinement step for PoseCNN [39], while [29] is presented as a refinement step for SSD6D [20], and DPOD [41] is presented along with a refinement step. In each case a synthetic image of the target object is rendered using the initial pose estimate and a 3D model of the target object. The key difference in these approaches is in the representation of the relative pose. DeepIM [27] utilise an 'untangled representation' in which the centre of the object is the centre of rotation, and translation is estimated in pixel space. Manhardt *et al.* [29] represent rotation as a unit quaternion and translation as a vector. DPOD [41] combines these approaches by estimating rotation relative to the object centre, and estimating translation as a vector. CosyPose [24] improve upon DeepIM by utilising a more recent feature detection network, and by estimating object poses in a multi-view scene.

B. Depth Estimation & Refinement

Research in monocular depth estimation can also be subdivided into three general areas: supervised estimation, semi-supervised estimation, and depth refinement.

Supervised depth estimation requires use of a ground truth depth map for network training, and is typically approached as a dense regression task [8], [25], [26], [2], [38]. Recent work in this area includes [10], which addresses the slow-convergence and poor spatial resolution issues inherent in some previous methods by implementing a deep ordinal regression network along with a space-increasing discretisation strategy. To enable wider application of depth estimation techniques [37] proposes a lightweight network suitable for embedded system applications.

Semi-supervised methods involve combining training data that includes ground truth annotation with training data that does not include ground truth information. Garg *et al.* [12] approach this problem by training with image pairs with a small, known displacement between images. The predicted depth map is then inverse-warped using the known relative displacement to reconstruct the second image. Similarly, [14] uses the concept of training with an image pair along with epipolar constraints to learn a disparity map between the image pair from which a depth map can be obtained.

Similar to object pose estimation, depth estimation is not always performed in a single step [7]. Recent approaches that focus on depth refinement include [13], [7]. Gidaris and Komodakis [13] approach depth-map improvement as a multi-learning problem with three modules; error detection, pixel-wise label replacement, and refinement. In contrast,

Durasov *et al.* [7] first learn a low resolution depth map which is iteratively refined with high-level features.

III. THE INNOVATION CNN

In this section we present a formulation for a general state estimation algorithm and apply this formulation to the problems of object pose estimation and depth estimation.

A. State Estimation with an Innovation CNN

Motivated by filtering and optimisation principles [1], we treat object pose and depth estimation as offline state estimation tasks. In this context a *state* is an internal representation of a system that is sufficient to fully define the future evolution of all system variables. The most general form of a state estimator can be written

$$\widehat{\mathbf{X}} = \mathcal{E}(\mathbf{V}), \tag{1}$$

$$\widehat{\mathbf{y}} = h(\widehat{\mathbf{X}}), \tag{2}$$

where $\mathbf{V}$ is the input and, in the problems considered, the estimator $\mathcal{E}$ is implemented as a CNN that estimates the state and the output function h estimates the variable of interest $\widehat{\mathbf{y}}$, eg. object pose. Due to the nature of CNNs the state is often a high dimensional regressor and the output function is a critical component to generate the low dimensional information typical in a robotics application.

The proposed approach involves taking the output state from an existing application-specific estimator network and iteratively refining this output using a state estimation framework. The class of stochastic gradient descent algorithms considered for refinement consist of a dynamical system that takes measurements from the true system as inputs and whose state converges to the true system state

$$\widehat{\mathbf{X}}(t+1) = \widehat{\mathbf{X}}(t) - \Delta(t), \tag{3}$$

$$\widehat{\mathbf{y}}(t+1) = h(\widehat{\mathbf{X}}(t+1)), \tag{4}$$

where $\Delta(t)$ is the innovation term that is chosen so as to drive the estimated outputs towards the true outputs. The task then becomes to learn an appropriate $\Delta(t)$ at each time, such that the error in the state estimation algorithm is iteratively driven towards zero.

Consider the standard L2-norm loss function

$$\Phi(t) = \frac{1}{2}\|\widehat{\mathbf{X}}(t) - \mathbf{X}\|_2. \tag{5}$$

The gradient of this function is

$$\begin{aligned}\nabla\Phi(t) &= \frac{1}{2}\nabla_{\widehat{\mathbf{X}}(t)}\|\widehat{\mathbf{X}}(t) - \mathbf{X}\|_2^2 \\ &= \widehat{\mathbf{X}}(t) - \mathbf{X},\end{aligned} \tag{6}$$

where $\mathbf{X}$ is the ground truth and $\nabla_{\mathbf{X}}$ denotes the gradient operator with respect to $\mathbf{X}$. Set

$$\Delta(t) = \alpha(t)\widehat{\nabla\Phi}(t), \tag{7}$$

where $\alpha(t)$ is a sequence of step-sizes that must be specified, and

$$\widehat{\nabla\Phi}(t) = \mathcal{F}(\mathbf{V}, \widehat{\mathbf{X}}(t)),$$

is an estimate of $\nabla\Phi(t)$ that is generated by the output of the Innovation CNN $\mathcal{F}$, given both the present estimate of state $\widehat{\mathbf{X}}(t)$ and the inputs $\mathbf{V}$ (eg. an RGB image).

B. State Estimation for Object Pose and Depth Refinement

For **object pose estimation** we choose PVNet [31] to be the baseline network that provides $\widehat{\mathbf{X}}(0)$. The output of PVNet is a unit vector field, with vectors pointing from each pixel to of a set of keypoints. The vectors are defined to be

$$\eta_{ij}^k = \xi^k - \xi_{ij}, \tag{8}$$

where ξ_{ij} denotes a 2D pixel with coordinates (i, j) within an image $\mathcal{I}$ with dimensions $M \times N$, and ξ^k represents the pixel location of keypoint $k \in \mathcal{K}$. The high dimensional regressor 'state' is the collection of unit vector fields

$$\mathbf{X}_{ij}^k = \frac{\eta_{ij}^k}{\|\eta_{ij}^k\|_2} \in \mathbb{R}^{2\times\mathcal{K}\times M\times N}. \tag{9}$$

The function h maps this vector field representation of keypoints to object pose via a RANSAC and uncertainty-driven PnP framework (EPnP), as discussed in [31].

For **depth estimation** we choose FastDepth [37] to be the baseline network that provides $\widehat{\mathbf{X}}(0)$. The output of FastDepth is a 1D per-pixel depth map, defined as

$$\mathbf{X}_{ij} \in \mathbb{R}^{M\times N}. \tag{10}$$

This depth map forms the state for the estimation problem for depth refinement, and the function h is an identity mapping as the output of the baseline network is the desired representation. For simplicity, the state is represented by $\mathbf{X}$ for both applications in the following sections.

C. Innovation CNN Framework

The Innovation CNN framework is developed to be applicable to a range of applications that are typically approached with an encoder-decoder network architecture. We begin by taking two copies of the baseline encoder-decoder network, eg. PVNet, FastDepth. We label the first network the *Innovation Estimator*, and modify the loss function to Eq. (13). We label the second network the *Estimate Autoencoder*, and modify the loss function to Eq. (11). We adapt the input later of the Estimate Autoencoder such that this network can receive a state estimate. We connect the two networks via skip connections to achieve the parallel network architecture shown in Fig. 2. The Innovation Estimator receives an image and returns an estimate of the gradient of the state $\widehat{\nabla\Phi}$. The Estimate Autoencoder receives a state $\widehat{\mathbf{X}}$ and returns an estimate of the state $\bar{\mathbf{X}}$. The Estimate Autoencoder is an auxiliary task aimed at allowing information from the state estimate to be injected into the Innovation Estimator.

For **object pose estimation** the baseline network is PVNet [31]. The input dimensionality of the Estimate Autoencoder is modified to accept a vector field state estimate.

For **depth estimation** the baseline network is FastDepth [37]. The input dimensionality of the Estimate Autoencoder is modified to accept a depth map state estimate.

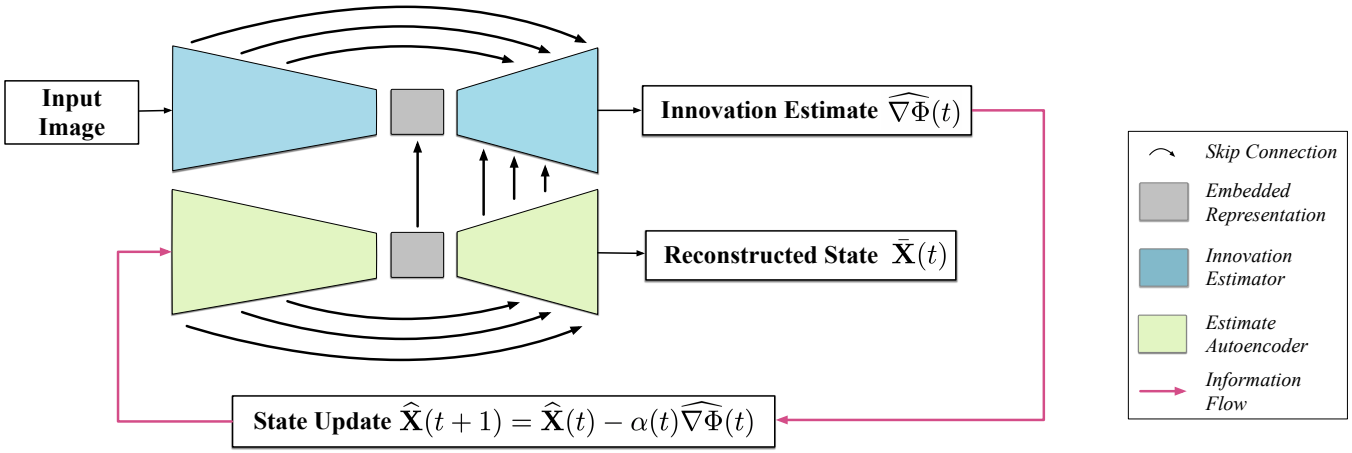


Fig. 2: Network Architecture. The input image is passed to the Innovation Estimator (top block), which estimates $\widehat{\nabla\Phi}(t)$. The current state estimate $\widehat{\mathbf{X}}(t)$ is passed to the State Autoencoder (bottom block). The output of the State Autoecoder is $\bar{\mathbf{X}}(t)$, which represents the same information as $\widehat{\mathbf{X}}(t)$. Using $\widehat{\nabla\Phi}(t)$ the state $\widehat{\mathbf{X}}^k$ is updated via gradient descent (Eq. (7)). Both encoders reduce input resolution to the same intermediate dimensionality (the 'embedded representation'), and skip connections are used to pass encoded state information to the Innovation Estimator.

D. Training Loss

The state loss is applied to the Estimate Autoencoder, and is defined by

$$\mathcal{L}_{\text{state}} = \sum_{(i,j)\in\mathcal{S}} \|\widehat{\mathbf{X}}(t) - \bar{\mathbf{X}}(t)\|_1, \tag{11}$$

where $(i,j) \in \mathcal{S}$ indicates that the pixel is within the segmentation mask. Note that for depth estimation the entire image is considered to be within the segmentation mask. The decoded state $\bar{\mathbf{X}}$ represents the same information as $\widehat{\mathbf{X}}$ and is used only for the purpose of training the autoencoder.

The state gradient loss is applied to the Innovation Estimator, and is defined by

$$\mathcal{L}_{\text{innov}} = \sum_{(i,j)\in\mathcal{S}} \|\widehat{\nabla\Phi}(t) - \nabla\Phi(t)\|_1 \tag{12}$$

$$= \sum_{(i,j)\in\mathcal{S}} \|\widehat{\nabla\Phi}(t) - (\widehat{\mathbf{X}}(t) - \mathbf{X})\|_1. \tag{13}$$

The loss is backpropogated after each iteration of the gradient descent defined by Eq. (7).

The loss function used to train the network is

$$\mathcal{L} = \mathcal{L}_{\text{innov}} + \gamma\mathcal{L}_{\text{state}}, \tag{14}$$

where γ is a tuning parameter.

E. Implementation Details

For both applications the network is trained for $T = 2$ iterations of Eq. (7) at each epoch, with a constant step size of $\alpha = 0.6$.

For **object pose estimation** we use a pretrained PVNet [31] to provide the initial estimate for each object. We use a batch size of 64, and down-sample training images by four to fit the learning algorithm on our machine. We compute 8 keypoints via farthest point sampling, from which object pose is obtained via uncertainty-driven PnP [31]. We also render 10,000 images and synthesis 10,000 images for each object. We employ data augmentation in the form of random cropping, resizing, rotation, colour jittering. An Adam optimizer [23] is employed with an initial learning rate of $1e^{-3}$, which is decayed to $1e^{-5}$ by a factor of 0.85 every 10 epochs. The learning hyperparameter in Eq. (14) is set to $\gamma = 10$. We train our network for 50 epochs. During testing, we employ Hough voting to localise keypoints, before using uncertainty-driven PnP to obtain object pose from keypoints. We set the step size to $\alpha = 0.01$ and perform iterations of Eq. (6) until the estimate converges.

For **depth estimation** we use a pretrained FastDepth network [37] to provide the initial estimate. We use a batch size of 8, and a Stochastic Gradient Descent with Momentum (SGDM) optimizer [35] is employed with a learning rate of 0.04, which is reduced by 10% every 5 epochs. A mask is applied to filter out missing ground truth values in the dataset. We train our model for 100 epochs. During testing, we set the step size to $\alpha = 0.4$ and perform iterations of Eq. (6) until the estimate converges. For both applications we consider the estimate to have converged once the gradient output of our Innovation CNN approaches zero.

IV. EXPERIMENTS

In this section we conduct experiments on widely used datasets for both object pose estimation and depth estimation and evaluate against standard metrics.

A. Datasets

We conduct experiments on two popular datasets for object pose estimation and one popular depth estimation dataset.

LINEMOD [16] is an object pose dataset consisting of 15783 images of 13 objects, with approximately 1200 instances for each object.

Occlusion LINEMOD [3] is a subset of LINEMOD images containing 8 objects, with each image containing multiple annotated objects under severe occlusion, thus

presenting a more challenging scenario for accurate pose estimation.

NYU Depth V2 [30] is a depth dataset that comprises of video sequences from a variety of indoor scenes, with a total of 1449 densely labeled pairs of aligned RGB and depth images.

B. Evaluation Metrics

We evaluate our approach using a standard metric for object pose estimation, and two standard metrics for depth estimation.

The **ADD** metric [16] computes the average 3D distance between the points of the 3D model under transformation from the ground truth and estimated pose respectively. For symmetric objects we use the similar **ADD(-S)** metric [39], where the mean distance is computed based on the closest point distance. We denote both metric as ADD(-S) and use the one appropriate to the object. A pose is considered correct if the average distance is less than 10% of the 3D model diameter. The reported metric is the percentage of poses that are considered correct.

Root Mean Squared Error (RMSE) is the most commonly used metric for evaluating depth estimation performance. RMSE computes the square root of the average of squared errors.

Mean Absolute Error (MAE) is another popular metric for depth estimation, that computes the average of the absolute errors.

We propose an additional metric, dubbed the **State Distance (SD)**, designed to quantify the difference between the estimated and ground truth state. This is used to provide an indication of whether the state estimate $\widehat{\mathbf{X}}$ is converging to the true state $\mathbf{X}$. This metric is defined by

$$\text{SD} = \frac{1}{s} \sum_{(i,j)\in\mathcal{S}} \|\widehat{\mathbf{X}}(t) - \mathbf{X}\|_2, \tag{15}$$

where s is the number of pixels within the segmentation mask. We make use of this metric in Fig. 6.

C. Comparison to Baseline Methods

Object pose estimation results in terms of the ADD(-S) metric on the LINEMOD and Occlusion LINEMOD datasets are presented in Table I. Depth estimation results in terms of the RMSE and MAE metrics on the NYU Depth V2 dataset are presented in Table II.

Our network generates estimates that lead to a higher average performance on both applications, compared to the baseline networks. Specifically, for object pose estimation we achieve a performance increase of 4.20% and 8.12% respectively compared to the performance of the baseline network, in terms of the ADD(-S) metric.

TABLE I: Performance comparison on the **LINEMOD** and **Occlusion LINEMOD** datasets in terms of the **ADD(-S)** metric.

	LINEMOD		**Occlusion LINEMOD**	
	Baseline	**Ours**	**Baseline**	**Ours**
ape	43.62	64.76	15.81	26.41
benchvise	99.90	99.90		
cam	86.86	92.58		
can	95.47	96.56	63.30	61.31
cat	79.34	82.83	16.68	19.88
driller	96.43	97.52	65.65	70.10
duck	52.58	63.31	25.42	31.99
eggbox	99.15	99.32	50.17	49.44
glue	95.66	96.91	49.62	51.16
holepuncher	81.92	82.20	39.67	42.34
iron	98.88	99.31		
lamp	99.33	99.42		
phone	92.41	94.22		
average	**86.27**	**89.92**	**40.77**	**44.08**

TABLE II: Performance comparison on the **NYU Depth V2** dataset.

	RMSE $\downarrow$	**MAE** $\downarrow$
Baseline	0.5868	0.4338
Ours	**0.5599**	**0.4127**

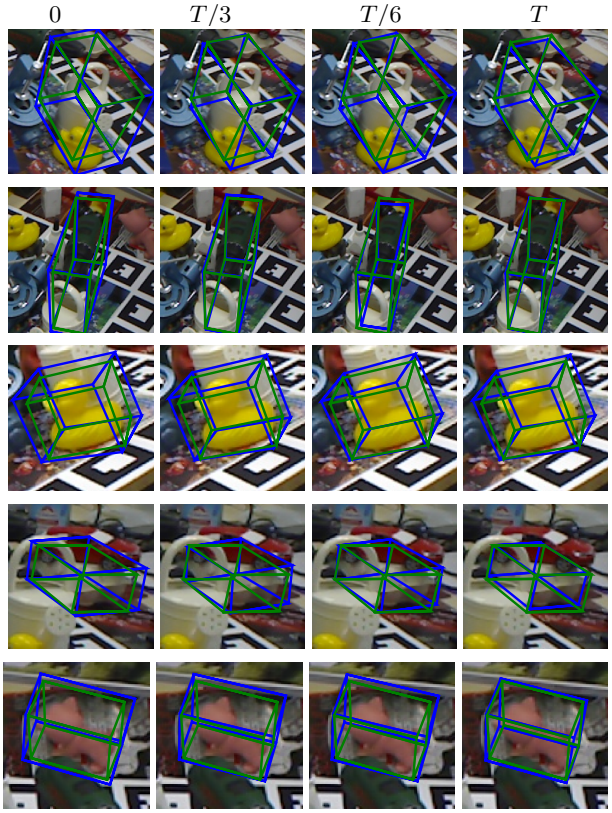


Fig. 3: Iterative pose refinement on example objects in the **Occlusion LINEMOD** dataset. The initial pose estimate is shown in the left ($t = 0$), and the final pose estimate is shown on the right ($t = T$), with equidistant intermediate poses shown in-between. Green bounding boxes represent ground truth poses and blue boxes represent our results.

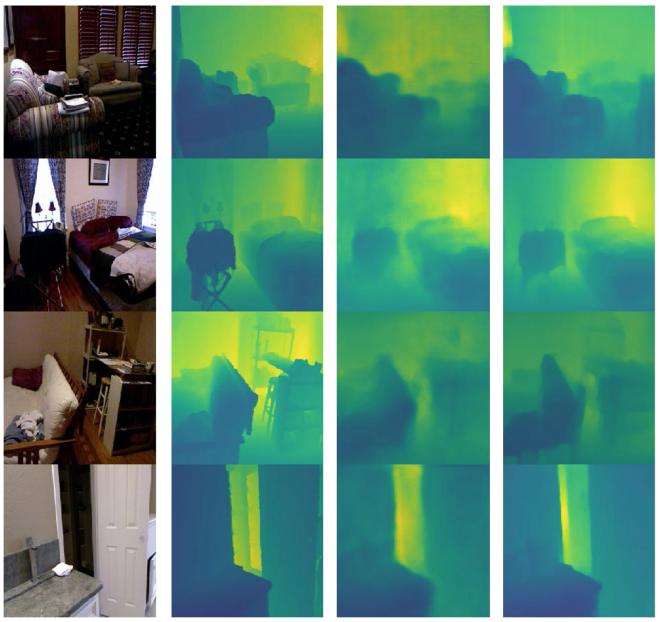


Fig. 4: Visualisation of example depth map. (a) input image, (b) ground truth, (c) baseline model, (d) our result.

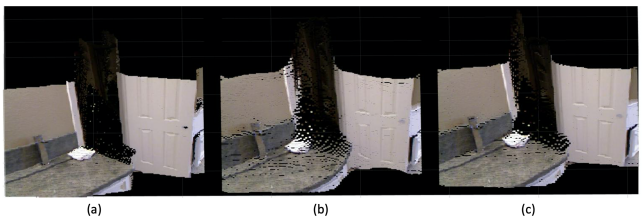


Fig. 5: Visualisation of example depth point cloud. (a) ground truth, (b) baseline model, (c) our result.

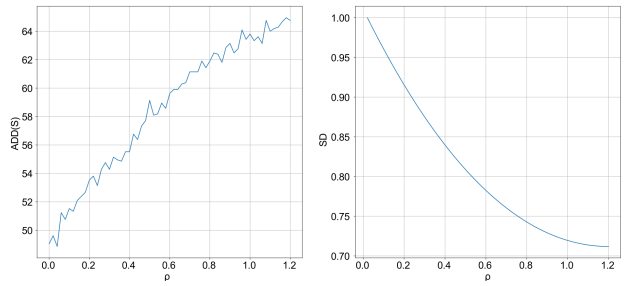
We record the largest improvement for objects for which the baseline network provides a relatively poor initial estimate. Such objects include the relatively texture-less 'ape' and 'duck' and all objects in the Occlusion LINEMOD dataset. On the ape and duck objects we improve baseline estimates by 21.14% and 10.73% respectively in terms of the ADD(-S) metric.

For depth estimation we achieve a performance increase of 4.58% and 4.86% in terms of the RMSE and MAE metrics respectively, relative to the baseline network. Qualitative results in Fig. 4 and Fig. 5 indicate that our framework is particularly effective at improving depth estimates at object edges and finer details such as chair legs.

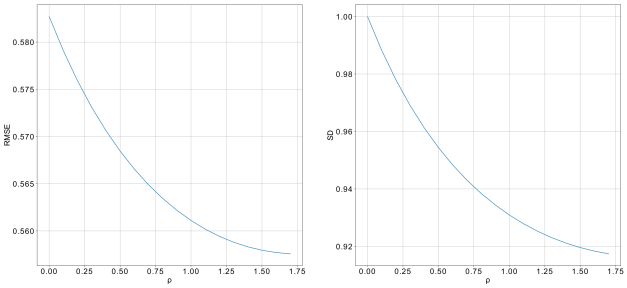
Fig. 6 illustrates the iterative improvement and convergence of the state distance SD obtained for both applications. In this figure, the ADD(-S), RMSE, and SD metrics are plotted as a function of the *interpolation distance*,

$$\rho = \alpha(t)T. \tag{16}$$

The interpolation distance quantifies how far we have iterated



(a) **Object Pose Estimation (Ape)**, $\alpha = 0.01$, $T = 120$.



(b) **Depth Estimation**, $\alpha = 0.1$, $T = 20$.

Fig. 6: Iterative improvement on the **ADD(-S)** and **RMSE** metrics and corresponding percentage decrease in the State Distance (**SD**) for object pose and depth refinement respectively, plotted against the interpolation distance ρ (Eq. (16)).

from the initial estimate. This figure indicates that the iterative stochastic gradient descent acts as expected to refine the state estimate and converge to a local minima (right-hand 'SD' plots). The primary metrics, ADD(-S) and RMSE, for the two problems are also shown to be optimised with the state convergence (left-hand plots). In this figure the step size $\alpha(t)$ was chosen smaller than was used to obtain the experimental results in order to visualise the evolution of the state distance and application-specific metrics. For experiments undertaken the same overall behaviour was obtained in two or three steps of the stochastic gradient descent with a larger $\alpha(t)$.

V. CONCLUSIONS

In this paper we present a general framework capable of improving initial estimates in multiple applications. The framework can be implemented for a given application by taking a baseline network developed for the task and adapting the network architecture and loss function, and reinjecting previous state information. The framework estimates the gradient from an initial state prediction to the true state. This gradient is applied iteratively in an SGD framework, greatly reducing the difficulty of estimating a state in a single forward pass. Experiments on widely used datasets demonstrate that we improve initial estimates significantly for object pose estimation and depth estimation.

REFERENCES

[1] BDO Anderson and JB Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, NJ, 1979.

[2] Shariq Farooq Bhat, I. Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. *ArXiv*, abs/2011.14141, 2020.

[3] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *ECCV*. Springer, September 2014.

[4] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[5] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox. Self-supervised 6d object pose estimation for robot manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3665–3671, 2020.

[6] S. C. Diamantas, A. Oikonomidis, and R. M. Crowder. Depth estimation for autonomous robot navigation: A comparative approach. In *2010 IEEE International Conference on Imaging Systems and Techniques*, pages 426–430, 2010.

[7] N. Durasov, M. Romanov, V. Bubnova, P. Bogomolov, and A. Konushin. Double refinement network for efficient monocular depth estimation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5889–5894, 2019.

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[9] R. Fletcher. *Practical Methods of Optimization; (2nd Ed.)*. Wiley-Interscience, USA, 1987.

[10] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[11] Sergio Garca, Mara Guilln, Rafael Barea, Luis Bergasa, and Eduardo Molinos. Indoor slam for micro aerial vehicles control using monocular camera and sensor fusion. 05 2016.

[12] Ravi Garg, BG Vijay Kumar, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016.

[13] Spyros Gidaris and Nikos Komodakis. Detect, replace, refine: Deep structured prediction for pixel wise labeling. 12 2016.

[14] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[15] Daniel Groos, Heri Ramampiaro, and Espen Ihlen. Efficientpose: Scalable single-person pose estimation. *Applied intelligence*, 2020.

[16] Stefan Hinterstoisser, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes.

[17] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[18] HyeokHyen Kwon, Yu-Wing Tai, and S. Lin. Data-driven depth map refinement via multi-scale sparse representation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 159–167, 2015.

[19] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[20] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1530–1538, 2017.

[21] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015.

[22] Wan-Soo Kim, Dae-Hyun Lee, Yong-Joo Kim, Taehyeong Kim, Won-Suk Lee, and Chang-Hyun Choi. Stereo-vision-based crop height estimation for agricultural robots. *Computers and Electronics in Agriculture*, 181:105937, 2021.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[24] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[25] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016.

[26] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019.

[27] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. *International Journal of Computer Vision*, 128(3):657678, Nov 2019.

[28] Zhigang li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *ICCV*, 10 2019.

[29] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in RGB. *CoRR*, abs/1810.03065, 2018.

[30] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[31] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019.

[32] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3848–3856, 2017.

[33] Mattia Rossi, Mireille El Gheche, Andreas Kuhn, and Pascal Frossard. Joint graph-based depth refinement and normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[34] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations, 2020.

[35] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[36] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. *CoRR*, abs/1711.08848, 2017.

[37] D. Wofk, F. Ma, T. Yang, S. Karaman, and V. Sze. Fastdepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108, 2019.

[38] Y. Wu, V. Boominathan, H. Chen, A. Sankaranarayanan, and A. Veeraraghavan. Phasecam3d learning phase masks for passive single view depth estimation. In *2019 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12, 2019.

[39] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes, 2017.

[40] Xin Yu, Zheyu Zhuang, Piotr Koniusz, and Hongdong Li. 6dof object pose estimation via differentiable proxy voting regularizer. In *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*. BMVA Press, 2020.

[41] Sergey Zakharov, Ivan S. Shugurov, and S. Ilic. Dpod: 6d pose object detector and refiner. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1941–1950, 2019.