T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ON THE REGULARIZATION OF 3D OBJECT POSE ESTIMATION

MUHAMMET ALI DEDE

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER ENGINEERING

GEBZE

2022

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ON THE REGULARIZATION OF 3D OBJECT POSE ESTIMATION

MUHAMMET ALI DEDE

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER ENGINEERING

THESIS SUPERVISOR

ASSIST. PROF. DR. YAKUP GENÇ

GEBZE

2022

T.C.

GEBZE TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

3 BOYUTLU NESNE POZ KESTIRIMINDE REGÜLARIZASYON

MUHAMMET ALİ DEDE

BİLGİSAYAR MÜHENDİSLİĞİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI

DR. ÖĞR. ÜYESİ. YAKUP GENÇ

GEBZE

2022

GEBZE TEKNİK ÜNİVERSİTESİ	**DOKTORA JÜRİ ONAY FORMU**

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 07/07/2022 tarih ve 2022/34 sayılı kararıyla oluşturulan jüri tarafından 28/07/2022 tarihinde tez savunma sınavı yapılan Muhammet Ali Dede'nin tez çalışması Bilgisayar Mühendisliği Anabilim Dalında DOKTORA tezi olarak kabul edilmiştir.

JÜRİ

ÜYE (TEZ DANIŞMANI)	: Dr. Öğr. Üyesi Yakup Genç
ÜYE	: Prof. Dr. Yusuf Sinan Akgül
ÜYE	: Doç. Dr. Habil Kalkan
ÜYE	: Doç. Dr. Ayşe Betül Oktay
ÜYE	: Dr. Öğr. Üyesi Tarkan Aydın

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ...../...../..... tarih ve ...../..... sayılı kararı.

İMZA/MÜHÜR

SUMMARY

6-DoF Pose estimation is finding the rotation and translation of an object to a preferential frame of reference. With the help of Neural Networks, recent pose estimation algorithms show significant improvements over the classical ones. Especially in complex scenes where challenging illumination, occlusion, and scene clutter impair the algorithm performance. However, the gain in performance carries a computational cost and expensive hand-annotated real-world data. Moreover, prominent algorithms require larger neural models to raise the bar, thus requiring more expensive training data. This thesis investigates the possibility of easing the computational burden and dependency on big data. We propose three different methods to address the issue. Our first model uses synthetically generated data to remove dependency on real-world data for pose estimation with a light-weight neural network. The second method incorporates aspect-classification and dense key-point estimation processes and recovers the object pose using robust algorithms. Our final study proposes a new curriculum-like training regime. This curriculum takes advantage of synthetic and real-world data and the behavior of neural networks against adversarial and noisy(perturbed) data. We conduct our experiments on public datasets to validate our proposed methods. Results demonstrate that the proposed methods perform comparably to the state-of-the-art algorithms while speeding up the inference time significantly.

Key Words: Object Pose Estimation, Deep Learning, Constraint Satisfaction.

ÖZET

6-DoF Pose tahmini, bir nesnenin tercih edilen bir nesneye konumunu ve rotasyonunu bulmaktır. Yapay sinir ağları yardımı ile geliştirilen sinir ağları klasik modellere göre oldukça başarlı olmaktadır Özellikle zorlu aydınlatma, engellenme ve sahne dağınıklığının görüntüyü bozduğu karmaşık sahneler de algoritmaların performanıs dikkat çekici hale gelimiştir. Bununla birlikte, performanstaki kazanç bir hesaplama maliyeti taşır. Özellikle elle etiketlenmiş verinin az ve pahalı olması ve önerilerin daha yüksek başarım için daha büyük modeller kullanması Bu algoritmaların uygulanabileceği platformları kısıtlamaktadır. Bu tez çalışmasında, hesaplama yükünü ve büyük veri bağımlılığı hafifletme olasılığını araştırılmıştır. Bu sorunları çözmek için üç farklı yöntem önerilmiştir. İlk yöntem, bağımlılığı ortadan kaldırmak için sentetik olarak oluşturulmuş verileri kullanırak daha küçük modeller kullanılabilir mi sorusunua cevap aramaktadır. İkinci yöntem, aspect sınıflandırması ve yoğun anahatar nokta tahminlemesi kullanarak nesne pozunu bulmaya çalışır. Son çalışmamızda ise , müfredat benzeri yeni bir eğitim rejimi önerilmektedir. Bu müfredatı oluşturulurken, sentetik ve gerçek dünya verilerinden ve yapay sinir ağlarının davranışlarından yararlanır. Önerilen yöntemlerimizi, diğer araştırmacılarında kullandığı için halka açık veri kümeleri test ettik. Elde ettiğimiz sonuçlar, önerilen yöntemlerin poz kestirim algoritmalarını hızlandırırken, en son teknolojiye sahip algoritmalarla karşılaştırılabilir bir performans sergilediğini gösteriyor.

Anahtar Kelimeler: Nesne Poz Kestirimi, Derin Öğrenme, Kısıtl sağlanması.

ACKNOWLEDGEMENTS

What a journey! What fun! If someone told me that I would be studying for a doctoral degree in 10 years again, I would probably think that someone, is going to hold a gun to my face. Yet I am there again, close to a degree. Frankly, I do not fully comprehend its purpose. Firstly, I would like to thank Allah, for the things he gave and the things he did not. He is behind all of this. I thank my advisor and Dr. Yakup Genç. It was an exceptional privilege to be working with a man who has incredible intellectual prowess and dedication. His guidance is the sole reason I could finish this herculean task. I also thank Prof. Dr. Yusuf Sinan Akgül for many, many things. But first and foremost, I would like to thank his patience with me. I will always cherish our conversations about anything related to us humans and our stories. I also thank Dr. Ayşe Betül Oktay for being there when needed. I would also like to thank all my friends and colleagues at Gebze Technical University who showed patience and understanding towards me. I also thank Dr. Erchan Aptoula for his contribution to other studies unrelated to the thesis. I applaud his dedication and tenacity. I would like to thank my family for everything. Their unconditional love and support held me together, whenever I needed it. Finally, I like to thank my girlfriend, who has shown me that happiness is not an elusive rare occurrence. It is a state of mind when you become a part of something different.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS and ACRONYMS

Abbreviations and Acroynms		**Explanations**
ADAM	:	Adaptive Momentum
ADD	:	Average Distance
ADD(S)	:	Average Distance with symmetry
ANN	:	Artificial Neural Network
CNN	:	Convolutional Neural Network
CPU	:	Central Processing Unit
GPU	:	Graphics Processing Unit
DoF	:	Degree of Freedom
FFT	:	Fast Fourier Transform
GAN	:	Generative Adversarial Network
IoU	:	Intersection over Union
NN	:	Neural Network
PnP	:	Perspective N point
RANSAC	:	Random Sample Consensus
RGB	:	Red-Green-Blue
ReLU	:	Rectified Linear Unit
SIFT	:	Scale Invariant Feature Transform

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

At its core, 3D computer vision is about sensing and inferring Geometry which we intrinsically use to interact with our surroundings. The sensing part is acquiring external signals and storing them in a structured representation. The inference, where computer scientists usually focus, is finding good representations and efficient algorithms to solve problems. This study focus on 6-Degrees of Freedom (6-DoF) pose estimation. The pose estimation problem asks about finding relative rotation and translation of an object to a canonical frame of reference. To this end, pose estimation has attracted significant interest from computer vision communities, since it is an essential part of Augmented Reality, Robotics, and autonomous driving. Prior approaches in computer vision solve this problem by extracting low-level features, and contours and matching them using a robust optimization algorithm. These methods generate promising results in simple structure scenes with feature objects. However, they fail to recover the pose under challenges like occlusion, scene clutter, or even texture-less objects. Like many computer-vision tasks, pose estimation also took its share with the rapid advancements in deep learning. Current learning-based methods outperform traditional ones in every metric. They demonstrate a powerful tool to work The central theme of this thesis is to build efficient and reliable techniques and notions for solving the 6-DoF pose estimation problem. In turn, this could be leveraged to construct successful applications. Researchers have studied these problems well and accumulated knowledge across decades. Yet, with the re-invention of deep learning, most of this valuable knowledge has started to fade away. In our thesis, we re-visit these problems from an optimization and constraint perspective where real-world annotated data is scarce and hard to obtain. Like most contemporary research, We heavily rely on the power of deep learning in our studies, yet we also utilize classical optimization techniques and notions. We believe, this union of old and new, results in simple and elegant solutions for pose estimation.

1.1. Motivation

As mentioned previously, contemporary pose estimation methods heavily depend on deep learning. Some studies rely on neural networks for end-to-end pose inference, while others estimate the location of key-points like object corners and random points on the object surface. Then, a robust PnP(Perspective-N-Point) algorithm is applied. Moreover, recent studies even use neural networks to estimate surface normals and depth information to provide information about the object geometry. Several studies also include a post-refinement stage where the predicted pose is crudely rendered and compared to the original image. These methods overcome the limitations of traditional pose estimation techniques by integrating deep learning. However, we observe several drawbacks in these studies.

Firstly, these methods heavily rely on large training datasets. These datasets for pose estimation require specialized hardware, a controlled environment, and hand labeling. While this may not pose a problem for competitive research, it impedes the development of industry-ready applications.

Additionally, we observe those ever-growing neural architectures to achieve better results in this particular task. It is well-known that neural networks do not suffer from the curse of dimensionality as long as there is data to be consumed. Keeping this in mind, researchers propose complex neural models to achieve state-of-the-art resultsPeng et al. [2019]; Akgul et al. [2016]. On the other hand, these large models exert a heavy computational burden. Hence, this hinders the applicability of the algorithms on multiple platforms. Additionally, several studies develop ad-hoc refinement procedures to increase the accuracy of existing approaches. Yet this also increases the complexity of the already heavy-weight approaches.

Finally, When analyzed in depth, these methods look like they are memorizing a variety of poses of the object. Provided that one has such diversity in the training data, a large enough network can learn a global semantic representation for features to be used estimation process. Yet with the lack of diversity, models tend to memorize the

training data and perform poorly in the out-of-distribution inference samples (Bias).

1.2. Overview of Methods and Contributions

Previously mentioned problems are real-world challenges that affect the quality of products. The thesis presents algorithms and notions that can significantly improve the performance of pose estimation applications and helps to solve the problems with elegant solutions. Our proposal also minimizes the dependency on specialized hardware (GPU, Accelerators) by employing low cardinality architectures. These architectures expand the applicability of available solutions to computationally limited devices. While we will present the detail of each study in their corresponding section, We would like to summarize our contributions case by case.

- In our first study, we construct an ultra light-weight neural network to directly estimate rotation and translation matrices. In our previous experiments, we noted that rotation matrices are harder to predict compared to translation vectors. To alleviate this, we parameterized the loss function and introduced an orthonormalization constraint on the rotation matrix as a regularizer. These changes enabled faster convergence in training and an observable drop-in validation loss.
- At the heart of this thesis, we introduce an aspect classification mechanism. In this study, we use a lightweight neural network for key-point estimation and object aspect classification. Then depending on the object aspect, our algorithm selects several key points from a canonical frame. Finally, we perform PnP- Ransac over a small set of selected key points. The proposed approach performs comparably to the state-of-the-art while consuming little memory and running on a mobile device in real-time.
- In the last study, we address the desperate need for data. We carefully construct a curriculum strategy. Our curriculum strategy heavily relies on synthetic data, online augmentation, Adversarial examples, and visual attention. We divide the training process into what we name ”chapters” where each chapter has its data-

sources learning rate policy. We tested the proposed method on two different well-known pose estimation architectures. Our empirical studies suggest a significant improvement, especially under challenging conditions like occlusion and scene clutter.

1.3. Other researches carried out during Doctoral studies and Outline

In addition to the Pose Estimation problem, we have been conducting research on theoretical and applied aspects of Deep Learning. Firstly, we have introduced two new ensemble methods and a training strategy similar to annealingDede et al. [2019b]. In another study, we proposed an algorithm for estimating additional spectra of earth images from aerial images. We have suggested a deep learning-based iterative approach on select datasets and showed that generated spectral images are very similar to the ground truthDede et al. [2019a]. During these studies, we heavily collaborated with Dr. Erchan Aptoula.

We have studied the representation efficiency of deep neural models. More specifically, in deep ReLU architectures, a significant number of neurons die out[1] during training. We have reduced the number of dead neurons by randomly perturbing the weights. Despite having minimal gains, the proposed algorithm was very slow due to batch calculations and memorization of activations. In a side study, We have explored the idea of surgically removing specific pathways of a neural graph and attach to another network to provide additional abilities. We have conducted experiments on the effects of adverse samples on training. We found that a subset of generated samples is architecture-independent. In other words, a neural network mispredicts these samples even if they do not target that architecture. During this study, we have also observed that incorporating these samples into the training data hampers the training process. We currently focus on neural rendering and implicit representations of the 3D geometry of scenes.

[1] Their output becomes zero and terminates gradient flow.

Finally, we have studied on explainability of machine learning-based decisions on a financial problem. We have developed a perturbation-based analysis approach with neighbor-hood analysis. We assume that the decision surface is a *Riemannian manifold*, and we have projected the feature space onto the decision space. We explored the topology of the decision surface by tracking gradients to perturbations. We have constructed decision atlases for combinations of features[2]. This research provided visually rich and more plausible explanations, compared to the generic explainability studies. On the other hand, we have failed to generalize the idea when projection space becomes more complex on carefully constructed toy datasets.

Our thesis proposes multiple approaches for better solutions 6-DoF pose estimation problems, yet they are not incremental research. Each study tries to address distinct(yet overlapping) challenges of the DNN-based pose estimation pipeline. Hence each part chapter has its narrative, datasets, and background. The rest of this thesis is structured as follows: In the next chapter, we propose a very lightweight pose-estimation architecture with algebraic constraints on rotation matrices. Chapter 3 presents a detailed study about the joint estimation of object aspects and key points, which we use to recover the object pose with the help of a robust algorithm. In chapter 4, we discuss the advantages of curriculum-based training in pose estimation problems. We present an extensive ablation study on different architectures. In the final chapter, we conclude our thesis with discussions and future work.

[2]Up to three, due to visualization constraints

2. COMPUTER VISION REVISITED

This chapter describes some core concepts of fundamental computer vision. We aim to provide the reader with sufficient background on the subject to enable the readers to navigate comfortably through the chapters. We certainly know that a few pages of a summary will bring more questions to the table than it answered. We encourage the reader to study [Hartley and Zisserman, 2003; Forsyth and Ponce, 2002; Lowe, 1999]. This chapter covers the pinhole-camera model and algebraic representations. In the final pages, we present a shortened version of Multi-View Geometry and extensively used algorithms.

2.1. Pinhole Camera Model

The pinhole camera model describes the mathematical relationship between the coordinates of a point in three-dimensional space and its projection onto the image plane of an ideal pinhole camera, where the camera aperture is described as a point and no lenses are used to focus light.

Assuming an image is obtained by a pinhole camera and a real-world point $(\mathbf{Q_x}, \mathbf{Q_y}, \mathbf{Q_z})^{\mathbf{T}}$ corresponds to the point $(\hat{\mathbf{Q}}_\mathbf{x}, \hat{\mathbf{Q}}_\mathbf{y}, \mathbf{f})^{\mathbf{T}}$ on the image plane (see Figure 2.1). We can derive the transformation by assuming homogeneous coordinates.

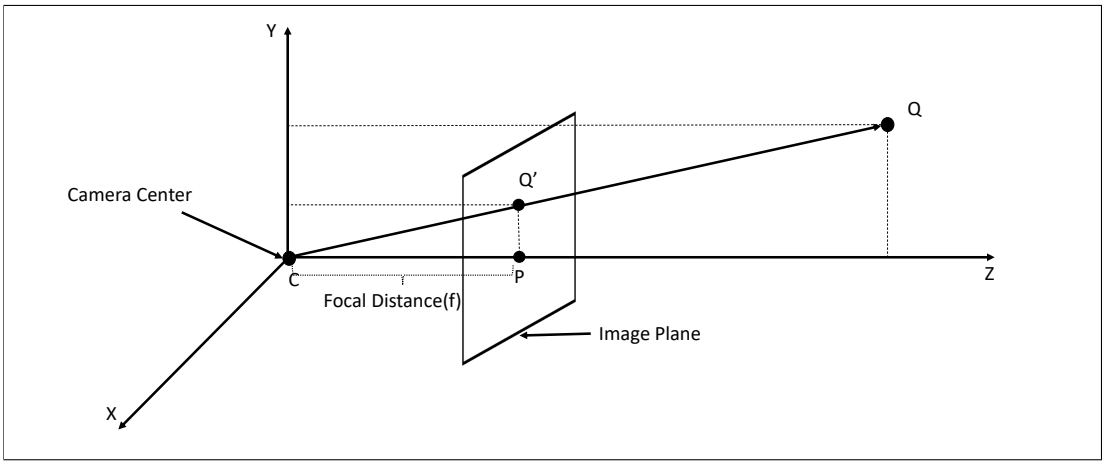

Figure 2.1: Pinhole camera model.

$$\begin{bmatrix} \hat{Q}_x \\ \hat{Q}_y \\ 1 \end{bmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} \tag{2.1}$$

Equation 2.1 is a over-simplification of camera projection and emphasizes on the focal length f. Real cameras are more complex and have many intrinsic variables. A more accurate representation:

$$\mathbf{K} = \begin{bmatrix} f_x & s & O_x & 0 \\ & f_y & O_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} I \end{bmatrix} \tag{2.2}$$

where f_x and f_y, are the focal length in pixel width and height the $\vec{x}$ and $\vec{y}$ axes. s denotes the $\vec{x}$ and $\vec{y}$ axes and (O_x, O_y) are the coordinates of the principal point in the image plane. The Matrix $\mathbf{K}$ is called Camera or The intrinsic Matrix sums up all the linear constraints of projective geometry. In addition to these linear constraints, there are other nonlinear such as lens distortion. While these distortions can be modeled and approximated, it is out of the scope of this thesis. These equations describe half of the story omitting the general transformation of a point in 3D-Space. In other words, this representation lacks the implicit 3D rotations and translations.

If $\mathbf{Q}$ is the coordinate of a point in the world coordinate frame, and $\mathbf{Q_{cam}}$ is the corresponding point in the camera coordinate frame, then we may write;

$$\mathbf{Q_{cam}} = \mathbf{R}(\mathbf{Q}\hat{\mathbf{C}}) \tag{2.3}$$

where $\hat{C}$ represents the coordinates of the camera. And $\mathbf{R}$ represents the Rotation Matrix in $\mathbb{R}^3$. We can re-write the equation in homogeneous coordinates by:

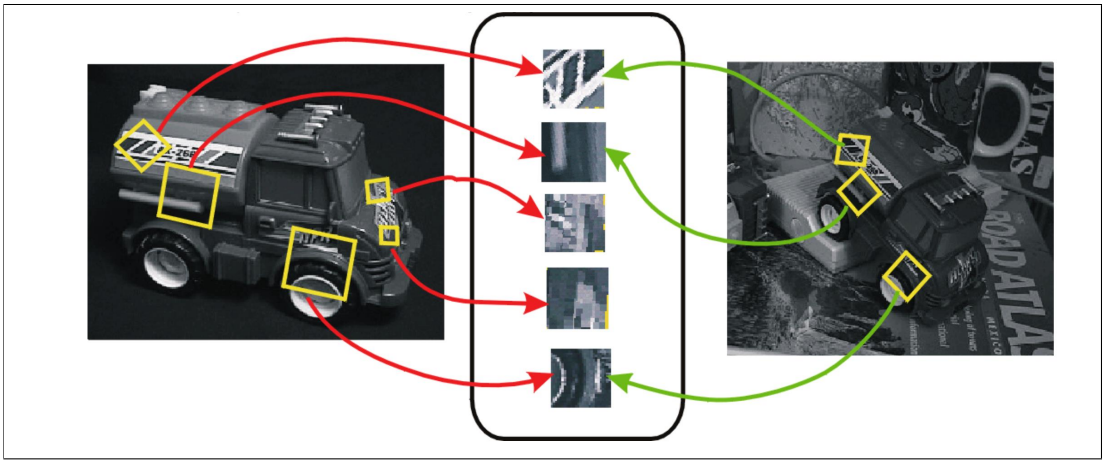

Figure 2.2: Correspondences between points

$$\begin{bmatrix} \mathbf{Q}_{cam[3\times1]} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{[3\times3]} & -\mathbf{R}_{[3\times3]}\hat{C} \\ \mathbf{0}_{[1\times3]} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{Q}_{[3\times1]} \\ 1 \end{bmatrix} \tag{2.4}$$

Combining 2.2 and 2.4 gives the general projective mapping for any camera.(Assuming no distortions). We usually omit the camera center from the equation and represent relative translation $-\mathbf{R}_{[3\times3]}\hat{C}$ as $\mathbf{t}$.

$$P = [KR \mid t] \tag{2.5}$$

where $\mathbf{P}$ is called the projection matrix.

2.2. Recovering 3D Pose

In the previous section, we have constructed an algebraic representation of camera geometry. Now, if the intrinsic parameters are known, the Projection matrix $\mathbf{P}$ has 6 degrees of freedom, 3 for rotation $\mathbf{R}$ and 3 for translation $\mathbf{t}$. Let's assume we want to recover the camera pose from an image. Solving the equation 2.5, we need to find matched points between the 3D world and the corresponding image like in figure 2.2. Figuring out which parts of one image correspond to the 3D world is not a trivial task. There are many challenges like scale, occlusion, illumination, and ambiguity. In

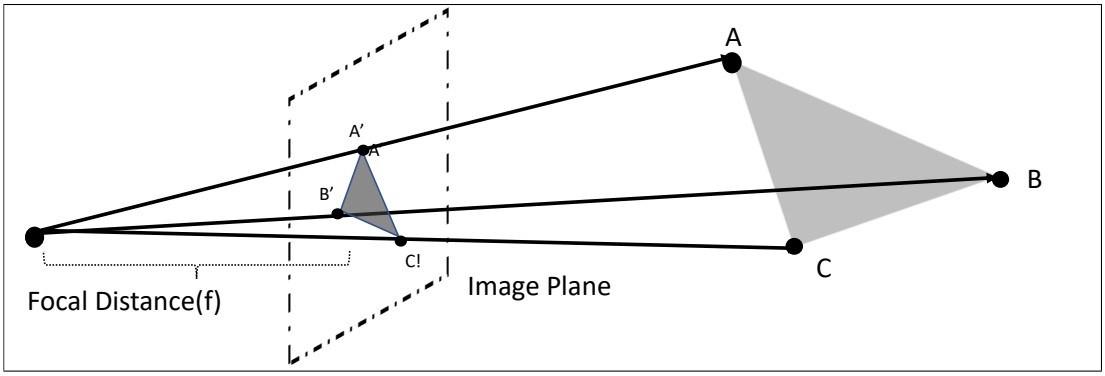

Figure 2.3: Geometric representation of 3 point projection. If the exact coordinates of correspondences are known, simple algebraic manipulation will yield the orientation and rotation of the camera.

classical computer vision, hand-crafted algorithms are used to find informative regions of an image. These regions usually rely on low-level features like edges, corners, blobs, ridges, or some combination of them [Lowe, 2004]. Using these features, we can find the position of a particular point in the image and map these to the known 3D world coordinates.

2.3. Perspective-n-Point Estimation

Now, we will examine how and to what extent can we recover the pose (Camera or an object) with our gained information.

Let's assume that, we have correspondences between an image and a 3D world for a set of image points $\mathbf{X_i}$, and world points $\mathbf{X_w}$, then there exists a unique Camera Matrix $\mathbf{P}$,

$$x_i = \mathbf{P}\, x_w \qquad \forall x_i \in X_i,\ \forall x_w \in X_w \tag{2.6}$$

Assuming cameras are calibrated, (even if not, there are several ways to find the intrinsic properties) and the correspondences are exact, solving the equation is a trivial task. Since we have learned that there are six unknowns (3 for rotations and 3 for translations) . Three correspondences as in figure 2.3 will suffice to recover the pose with simple algebra. Furthermore, efficient algorithms exist since the end of the

19th Century to solve the problem [Gao et al., 2003; Kneip et al., 2011; Haralick et al., 1991]. However, in real-life scenarios, there are several challenges to overcome. Firstly, exact correspondences are virtually non-existent due to imperfections, miscalculations, noise, and distortions. These results heavily affect the robustness and stability of the algorithms. Secondly, We usually gather more than three correspondence, (an order of magnitude larger). Additional matchings diminish the noise problem since larger point sets have redundancy that usually diminishes noise interference. However, correspondence sets also pose another problem: Computational burden, since inference should be performed on a much larger set. Keeping all this in perspective, we found out that our initial solution equation 2.6 was a naive approach. We now have a set of the equation that is highly over-determined and corrupted by some interference. In this case, any standard approach minimizes some algebraic (equation balancing) or geometric (back-projection or re-projection errors) error with some constraints[Lepetit and Fua, 2005; Lepetit et al., 2009; Terzakis and Lourakis, 2020]. Let $\mathbf{J}$ be the cumulative error

$$\mathbf{J} = \sum_i \mathbf{M}(x_i, \mathbf{R}\, X_i + \mathbf{t}) \tag{2.7}$$

where $\mathbf{x_i}$ and $\mathbf{X_i}$ are the corresponding image and world coordinate points. The cost function $\mathbf{M}$ is usually often employed with non-linear least square to iteratively refine an initial camera pose estimate.

2.4. Feature-Matching

Up until now, we have assumed that correspondences are already known. However, in a real-world application, we do not have that luxury. What we have is a noisy, over-determined, and unordered set of points that correspond to some regions of the image (See figure 2.4). Let's have a thought experiment and assume that $\mathbf{n}$ image points correspond to a subset of $\mathbf{m}$ real-world candidates. Broadly speaking, probability of finding the right matching for just one pair is $\frac{\mathbf{1}}{\mathbf{n}}$ and for the whole image point set it

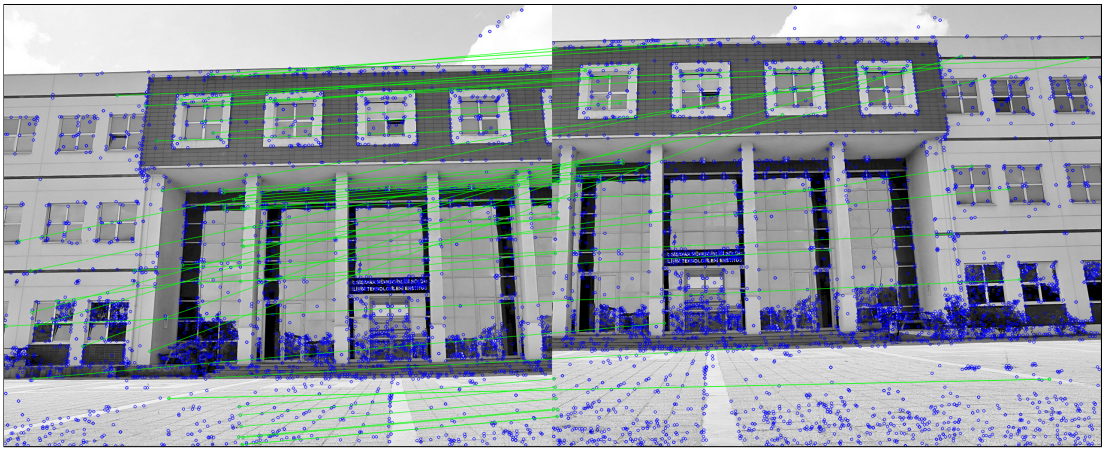

Figure 2.4: A crude point matching algorithm suffers from the immense amount of possible correspondences. Green Lines show the estimated correspondences. It can be observed that most of the proposed matchings are incorrect.

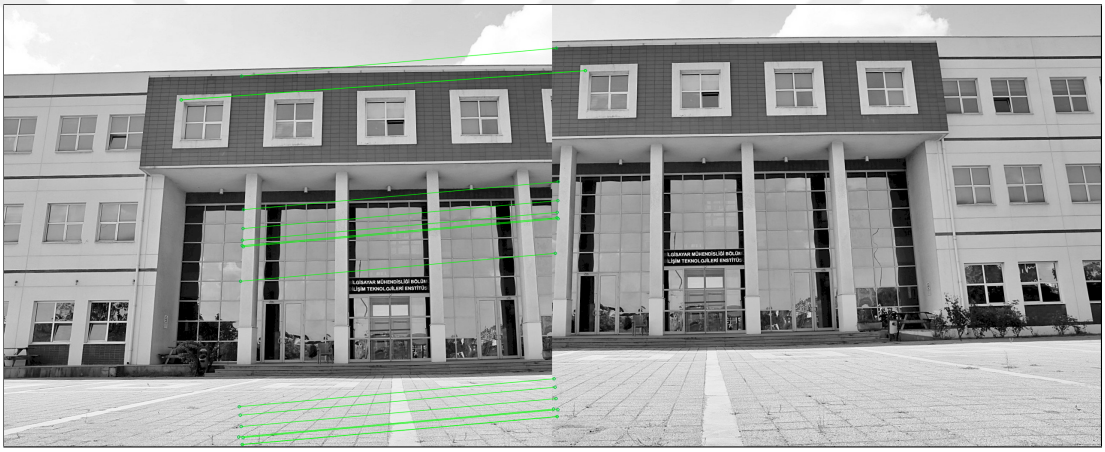

Figure 2.5: RANSAC culls most of the outliers based on the strongest hypotheses.

is $\frac{(\mathbf{n}-\mathbf{m})!}{(\mathbf{n})!}$[1]. Finding the best solution (still not exact; since we already introduced the noise and the distortions), requires an algorithm with complexity $O(m^{n-m})$. This is a computationally very challenging problem, especially for applications that require fast decision-making. Furthermore, the mismatched correspondences or *outliers* can disturb the estimation process and result in poor inference. Consequently, we have to determine which correspondences are real correspondences and which constitute outliers. In computer vision, what is often employed is an robust culling algorithm called **RAN**dom **SA**mple **C**onsensus (RANSAC) [Fischler and Bolles, 1981] algorithm. The idea behind this algorithm is simple. Given a set of matching points with enough

[1] Assuming $n > m$.

inliers, RANSAC randomly samples a sufficient amount of points to construct a model. The model validity is measured using the rest of the samples, if the samples are within some error threshold then they will be counted as inliers and the model will be rebuilt by this *Consensus Set*. This process is repeated several times and the hypothesis with the most support is deemed the best model. The intuition is that if one of the points is an outlier then the model will not gain much support. Thus, the model will be selected with the consensus of the inlier points. Figure 2.5 shows the effect of RANSAC on the previously estimated(see Figure 2.4) feature-matches. Unlike most optimization algorithms, RANSAC builds an initial model from a very small subset of samples. Then iteratively expands the models with consistent data, if possible. Furthermore, given enough iteration steps, RANSAC is guaranteed to find the best hypothesis albeit this solution might not be the most efficient one. While we present a summary of the algorithm here, Some questions like *What is a good inlier threshold, and How many times should we iterate? And if the outliers construct a valid consensus set.* are beyond this summary and they usually require a domain and application-specific expertise. The original algorithm is quite complex and heavily refers to itself, we present a summary from Multi-View Geometry [Hartley and Zisserman, 2003].

3. DIRECT POSE ESTIMATION FROM RGB IMAGES

In this chapter, A real-time monocular camera pose estimation algorithm is presented. Proposed model is a small convolutional neural network that is trained to directly estimated 6-DOF camera pose from an RGB image. Our model is designed to run on real-time devices with low memory and computation power. Our model can estimate the camera pose less than *1ms* while keeping accuracy comparable to the state-of-the art. This was made possible by employing geometrically sound loss functions and algebraic constraints. Furthermore, we introduce a new synthetic dataset for demonstrating the proposed methods capabilities.

3.1. Introduction

6D accurate pose estimation is an essential part of the computer vision research since it has important applications in robotics, navigation an augmented reality. This problem is not trivial; many problems like occlusion, illumination and dynamic background can interfere with the estimation. Like many other fields in computer vision, pose estimation also received its fair share of deep learning-based attention. Recent deep learning methods SSD6 [Tekin et al., 2018] and YOLO6 [Sock et al., 2018] is built upon well known object detector SSD [Kehl et al., 2017] and YOLO[Redmon et al., 2016]. These models focuses on estimating the 6D pose of an object, depending on success of 2D counterparts on bounding box estimation of given objects. However, estimating 6DOF pose is a much harder problem in RGB images. Variations in appearance, ambiguities and the lack of geometric information and depth [Sock et al., 2018] may complicate robust estimation.

This manuscript addresses recovering camera pose relative to a particular object from a monocular RGB image. This approach is analogous to object pose estimation since our research is focused on augmented reality (AR) perspective. In AR applications, estimating egocentric motion and pose relative to the scene is important as objects in

the scene usually stay stationary.

Camera pose estimation is an important part of Structure-from-Motion and image-based localization. This problem is traditionally well-studied[Forsyth and Ponce, 2002; Hartley and Zisserman, 2003] . Algebraically, given a set of correspondences between an image and its 3D model, the camera pose can be calculated by solving a six-degree polynomial. However, in practice, there are several problems with algebraic approaches. The first problem is finding reliable matching. SIFT[Lowe, 2004] and other gradient-based variants are good at finding such informative regions between the images. However, the output of such methods are usually noisy and requires robust methods like RANSAC. The second problem is these methods are local, try to understand very low level-features like corners and edges without paying attention to global structures in the image. To address such problems Deep Learning Based camera pose estimators such as PoseNet[Kendall et al., 2015; Walch et al., 2017], are usually applied onto large scale localisation. Following these works, new models are used to learn relative ego-motion[Melekhov et al., 2017] and compute pair-wise camera pose [Laskar et al., 2017], improve the context of features[Walch et al., 2017], and Bayesian Neural Networks[Kendall and Gal, 2017]. Main advantage of such probabilistic learning, they do not suffer from the curse of dimensionality, since they can learn to represent many important and complicated features in the image context. Furthermore, given enough diverse samples, they cope with changing environmental factors well. However, deep learning based approaches posses some drawbacks. First of all, they require immense amounts of labeled training data. Gathering and annotating such data requires lots of human effort and is expensive. Furthermore, this process is usually error prone and it results in contaminated training data that can impair the performance of the model. Secondly, neural networks require lots of computation power and memory. This is especially a drawback from AR perspective, since contemporary AR devices are mobile phones, which have limited computational and energy resources.

Our work tries to address such problems with minor contributions

- We present a new synthetic dataset with a class of objects(cars) and training pipeline.
- We make a distinction between the camera pose with respect to an object versus a class of objects using synthetic datasets.
- We demonstrate that our model can estimate the unseen poses of an object using seen samples of another object belonging to the same class.
- We constructed a small CNN architecture, which is designed to run on with computationally limited capabilities that run on real-time.
- We introduced an geometric constraint into loss function as a regularizer. Experimental study shows that our constraint reduces training time and slightly increases performance of our proposal on our dataset.

Following the literature, we have also tested capabilities of our method on competitive benchmark datasets and achieved comparable results while estimating the camera pose around 20 milliseconds.

3.2. Direct Estimation of Camera or Object Pose

Direct of pose estimation is to train network for estimating 3D translation and 3D rotation. In this scenario, training samples are fed to the network and trained with rotation and orientation vectors as ground truth. One way to perform the estimation is regressing directly the labels. Another approach is to turn this problem into a classification problem [Kehl et al., 2017] by discretizing the poses. Another strategy for pose estimation is using key-point based methods [Oberweger et al., 2018; Rad and Lepetit, 2017]. These methods use neural structures to propose key-points of the objects, and then use Perspective-n-Point algorithm to estimate the final pose. However such methods suffer from noisy outputs of due to occlusions in CNN's[Rad and Lepetit, 2017]. Recent studies [Peng et al., 2019] employ hybrid estimators that merge neural networks with RANSAC .

Given a monocular input image **I**, we aim to estimate the location **T** and orien-

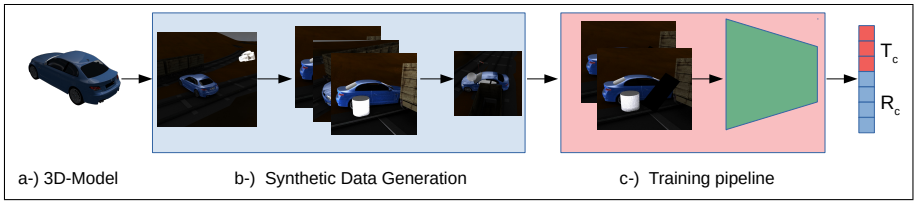

Figure 3.1: Direct estimation of camera pose using neural models using synthesized images.

tation **R** of the camera with respect to the object (see Figure 3.1). This is the same as estimating the object pose with respect to the camera. Algebraically, the imaging process is captured by Equation 3.1 where **K** is a 3×4 matrix capturing the camera internal parameters, $[x, y, z]^T$ is a point in object coordinate system and $[u, v]^T$ it's image and ρ is the protective depth.

$$\rho \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.1}$$

Camera pose and orientation can be represented as a 7 dimensional vector $\theta = [t, r]$ where t is a three-dimensional vector representing the pose or the translation of the camera. And r is a four-dimensional vector representing camera rotation as quaternions. As in [Kendall et al., 2015] we choose quaternions for representing rotations since they lie on a unit hyphersphere, any arbitrary four-dimensional vector represents a rotation provided they are normalized. Another advantage of this property is that, we can use this term as a constraint to help to train the network. Unlike Euler angles it is easier to apply a constraints on quaternions rather than euclidean angles that require complex orthonormalization computations.

Direct estimation of camera pose from an image is done by a neural model (see Figure 3.1). A custom neural model optimized for speed and small memory footprint is designed in order to run the model on devices that have limited computational capabilities. Proposed architecture (see Figure 3.2), down-scales the given image by

three consecutive 5x5 convolutions with strides of 2 and applies a normalization. Then, we construct a convolution block that consists of 3 densely connected [Huang et al., 2016] traditional 3x3 convolutions with a bottleneck 1x1 convolutions which is followed by an average pooling and a normalization layers. This is applied three times. The model is continued with two fully connected layers each 1024 neurons. Model output is a 7-dimensional vector in which the first three represent the location and the remaining four represents the rotation.

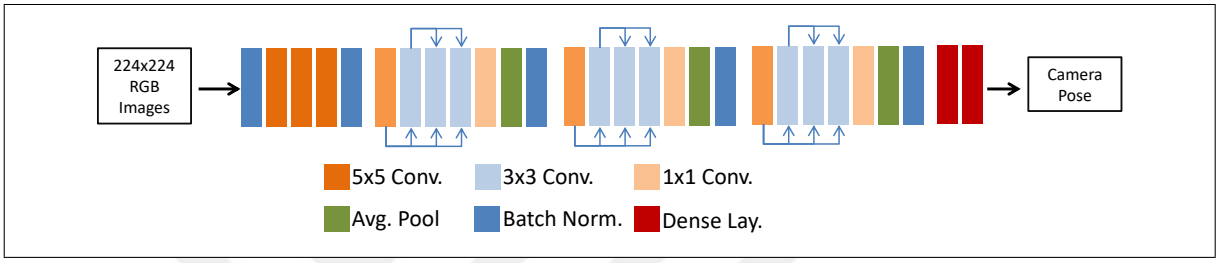

Figure 3.2: Proposed architecture for direct estimation of camera pose.

The model is trained through an optimization on a specially designed loss function. The proposed loss function is an amalgamation of different losses penalizing various estimations. The first part of the loss penalizes the difference between estimated position t_e and the ground truth t_g:

$$L_t = \|t_g - t_e\|_2 \tag{3.2}$$

This loss treats all three component of the position equally including the distance to the object. Our datasets have comparable values for lateral motion of the camera as well as the distance to the object.

The second loss is the simple difference between estimated rotation and actual rotations:

$$L_r = \|r_g - r_e\|_2 \tag{3.3}$$

Although not all the ranges of the rotation is covered in our training and test data due to limited access to a real world object, our loss yields again comparable values for the

four components of the rotation in quaternion representation.

The final loss is the projection error. Projection error is calculated by the euclidean distance between the actual image coordinates $[u_g, v_g]^T$ of a random set of points in the world and their estimated image coordinates $[u_e, v_e]^T$ (as given in Equation 3.4). During training we choose $n = 4$ random points to calculate a mean of the projection as the final projection loss:

$$L_p = \frac{1}{n}\sum_{i=1}^{n} \| \begin{bmatrix} u_g \\ v_g \end{bmatrix} - \begin{bmatrix} u_e \\ v_e \end{bmatrix} \|_2. \tag{3.4}$$

Once again, we do not distinguish between the locations of the points on image or object. Randomly choosing these points gives us enough variety, as the their placement on the object as well as their distribution in the image. Fixed set of points may create biases in this loss as they may lie on a singular setup (e.g., along a line) or they may map on the boundary of some images (effecting the range of values as well as being subjected to different camera distortions).

We also introduced the unit hyper-sphere constraint of the quaternions as a regularizer on the network. Kendall et al.[Kendall et al., 2015] mentioned about this fact and they decided not to use as part of optimization, believing that such constraints can impair model training. Furthermore, they observed that as the training advances, estimated pose comes very close to the ground truth and renders such additions unnecessary. However, our experiments show that such a constraint can be beneficiary to the overall training process. This will be discussed in detail in Section 3.4. We note that [Kendall et al., 2015] argues that as the training progresses, the estimated values of the camera rotation come close to the unit sphere of a quaternion. Instead of leaving this free, we add this constraint directly. However, this constraing:

$$Q = W^2 + i^2 + j^2 + k^2 = 1 \tag{3.5}$$

regularizes our network when used in this form:

$$\hat{Q} = e^{(Q-1)^2} \tag{3.6}$$

which as a simple partial derivative w.r.t. the parameters to optimized. Furthermore, it can be calculated using autograd of deep learning frameworks. As can be seen in Figure 3.3, the convergence properties of this loss behaves better than that of Equation 3.5.

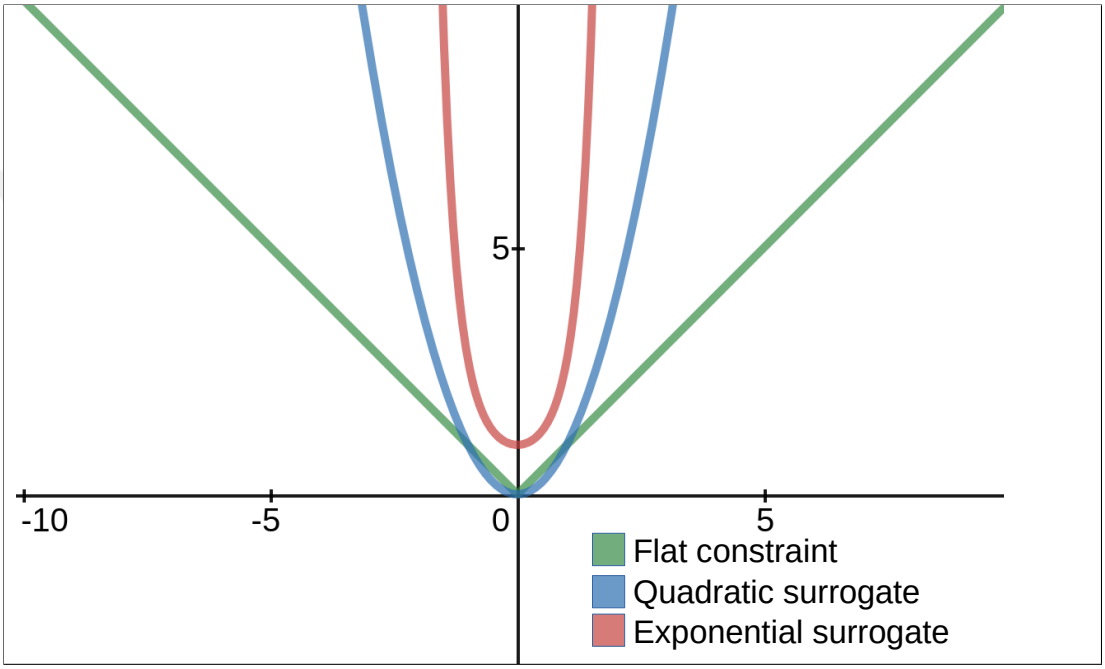

Figure 3.3: Typical convergence properties of Q vs its surrogates

We finaly combine all four lossed given in equations (3.2),(3.3),(3.4) and (3.6) into a single loss:

$$J = \alpha L_t + \beta L_r + \gamma L_p + \lambda \hat{Q}, \tag{3.7}$$

where α, β, γ and λ are hyper-parameters which are used to fine-tune and balance the overall training process. This combined loss function is differentiable and can easily be optimized by modern deep learning optimizers.

It should be noted that some portions of the loss may seem redundant. For example, the re-projection loss can represent both the translation and the rotation

losses together. However, stating these losses seperately may allow the optimization process to remove biases due to imperfections in the projection models, noises and other possible biases. And our experiments show that this improves the estimation accuracy significantly.

3.3. Datasets Used in Experimental Evaluation

It is well known that deep learning performs well on large datasets. However, producing such datasets are usually expensive or requires hand-annotated labels. We solved this problem by using generated 3D models. Given a 3D model, we have rendered an image with random object colours under different lighting conditions with random backgrounds.

We constructed a set of 3D models from open source communities. For each model, we have generated 600 images for training, 300 images for validation and final 1000 images with transparent backgrounds for testing. Rendered images have size the of 640x640 pixels. They are scaled down to 224x224 pixels during training. Rendering camera is randomly placed between two hemispheres with radius of r_1 and r_2 and oriented such that the target object is always visible within the image. For initial experiments these values are assigned 5 and 15 respectively. Additionally, we placed random 3D models between camera and the rendered object to simulate occlusion.

During dataset generation, camera hemisphere is divided into four quadrants along the x-axis and y-axis. 540 of the rendered training images come from three quadrants and 60 of them comes from the last one. We employed this strategy to test the interpolating capabilities of the neural network. Validation and test images are equally distributed among the quadrants.

We have also rendered pure silhouette of the object from the same camera perspective so that we can use IoU (*Intersection over Union*) score. This score choice is further discussed in the experiments section.

We have also demonstrated the performance of our method on *Cambridge Land-*

marks dataset [Kendall et al., 2015]. This dataset includes 5 large scale urban scenery with challenging conditions like pedestrians, vehicles and confusing environmental conditions like different lightning and weather conditions. For indoor comparison, we used *7Scenes Dataset* [Shotton et al., 2013]. This dataset contains 7 in-door scenes with various numbers of RGB-D images for each category. Although this dataset is designed for RGB-D re-localization, we use it to demonstrate the performance our proposed method. With the use of depth data, some of the difficulties can be alleviated. Our method will not make use of the depth data in our experiments. Also this dataset is also challenging for methods using image level features like SIFT, as it contains many ambiguous textureless areas.

3.4. Experimental Evaluations

3.4.1. Setup

We have pre-trained our model using PlacesNet[Zhou et al., 2018]. This dataset contains 1,803,460 training images with 365 categories. Each category has varying number of images from 3,068 to 5,000. PlacesNet dataset focuses on indoor and outdoor scenes. Alternatively ImageNet is used for pre-training networks in many applications [He et al., 2016]. However objects in ImageNet does not have the depth variations that we would like to see. The scene categories in PlacesNet has better depth variation which might be useful in our pose estimation problem.

Our model was implemented in TensorFlow[Abadi et al., 2016]. We used

Figure 3.4: Sample images from Cambrige scenes dataset.

Figure 3.5: Sample indoor images from 7scenes dataset.

Figure 3.6: Sample from our synthetic dataset.

Adam[Kingma and Ba, 2015] optimizer with a learning rate $5x10^{-4}$,which is halved for every 40 epochs. Overall training took 170 epochs and we employed early stopping to counter overfitting. Because of the limited GPU resources, batch size is selected to be 8. We also applied random geometric crops onto training images for both occlusion imitation and data augmentation(See Figure 3.1).

3.4.2. Hyper-parameter Selection

In experiments with our synthetic datasets, we selected α, β, γ, and λ as 3, 10, 20 and 8 respectively. α, β, and γ values are found using a grid search in the range [0,20] with 1 increments. Finding a healthy λ value is essential since selected loss approaches its minimum very slowly and many different configurations of quaternion may result in a similar regularization loss. In our experiments, we have hand tuned λ with less than 10 different tries. It can be argued that introduction of many hyper-parameters complicates training process, because balancing between angular loss and distance loss. Since the difference between losses may show large variation. Solving this problem requires domain knowledge and a search with many experiments. Furthermore, selected hyper-parameters may have to be updated under different scenarios and datasets. This problem is visited in a follow up article by Kendall et al. [Kendall and Cipolla, 2017]

and they used a geometric loss similar to our re-projection error again based on the fundamentals of multi-view computer vision geometry. They managed to improve their original performance [Kendall et al., 2015] while simplifiying the training process. However, in our experiments with synthetic dataset, we observed that, merging different losses with coefficients have a slight improvement on validation results.

3.4.3. Training Pipeline

During training, for each image batch we have selected random images from MS-COCO [Lin et al., 2014] dataset as background for images. Each synthetic image is overlayed on top of the choosen image. Thus, neural model almost sees a unique background for every rendered image. We have also employed a different occlusion strategy in training batches. We have applied random shaped crops in 20 percent of the images. While crop size depends on the shape that is assigned, for rectangular shapes, we randomly choose width and height between 10 and 40 pixels. For spherical crops, radius was again between 10 and 40 pixels. The cropped area is either filled with random uniform colors or we have cropped the area with same coordinates from the background and applied on top of the cropped region.

Table 3.1: A detailed dissection of Cambridge dataset.

Scene	Train img.	Test img.
King's College	1220	343
Old Hospital	895	182
St Mary's Church	1487	530
Shop Facade	231	103
Chess	4000	2000
Fire	2000	2000
Heads	1000	1000
Office	6000	4000
Pumpkin	4000	2000
Red Kitchen	7000	5000
Stairs	2000	1000
Synthetic Cars(ours)	6000	10000

Table 3.2: Performance of the proposed model in comparison to the state-of-the art.

Scene	Bayesian	LSTM	PoseNet*	Our Model
King's College	1.74m,4.06°	0.99m,3.65°	0.88m,1.04°	0.93m,2.64°
Old Hospital	2.67m,5.14°	1.41m,4.20°	3.20m,3.29°	2.18m,3.67°
St Mary's Church	2.11m,8.38°	1.18m,7.44°	1.57m,3.32°	2.11m, 6.8°
Shop Facade	1.25m,7.54°	1.52m,6.84°	0.88m,3.78°	1.16m,5.43°
Chess	0.37m,7.24°	0.24m,5.77°	0.14m,4.48°	0.26m,4.24°
Fire	0.43m,13.7°	0.34m,11.9°	0.27m,11.3°	0.43m,13.11°
Heads	0.31m,12.0°	0.21m,13.7°	0.17m,13.0°	0.21m,13.24°
Office	0.48m,8.04°	0.33m,8.08°	0.19m,5.55°	0.21m,5.98°
Pumpkin	0.61m,7.08°	0.37m,7.00°	0.26m,4.75°	0.53m,5.11°
Red Kitchen	0.58m,7.54°	0.58m,7.54°	0.24m,5.52°	0.24m,5.35°
Stairs	0.48m,13.1°	0.48m,13.1°	0.37m,12.4°	0.38m,12.1°
Synthetic Cars(ours)	-	-	0.18,1.19°	0.21,1.07°

3.4.4. Experiments and Results

During inference we normalize the estimated rotation by its length so that quaternion constraint is applied using $Q/\|Q\|$. We conduct experiments and make comparisons with the normalized estimations.

We show that proposed model is able to estimate camera pose effectively using only a fraction of the number of parameters used by PoseNet. In fact, the number of parameters are reduced from 22M to 2.4M as shown in Table 3.5.

Following [Kendall et al., 2015], we calculate median translation and rotation errors for all datasets and categories in Table 3.2. Additionally, we used IoU score for demonstrating the accuracy of tested models. There are multiple ways of measuring estimators performance. Although using such a score is nontraditional, we thought that using a simple intersection over a union of silhouette images can give both quantitative and qualitative nature of the comparison. It can be argued that our method has an inherent weakness, that different pose estimation can result in similar IoU scores. However, during our evaluation of predictions, we observed that even if the prediction is very close to the truth value, IoU may vary noticeably, since our objects have complex surfaces and cover a large area in the image. In order to calculate IoU score we rendered silhouette images of from both prediction and original pose.

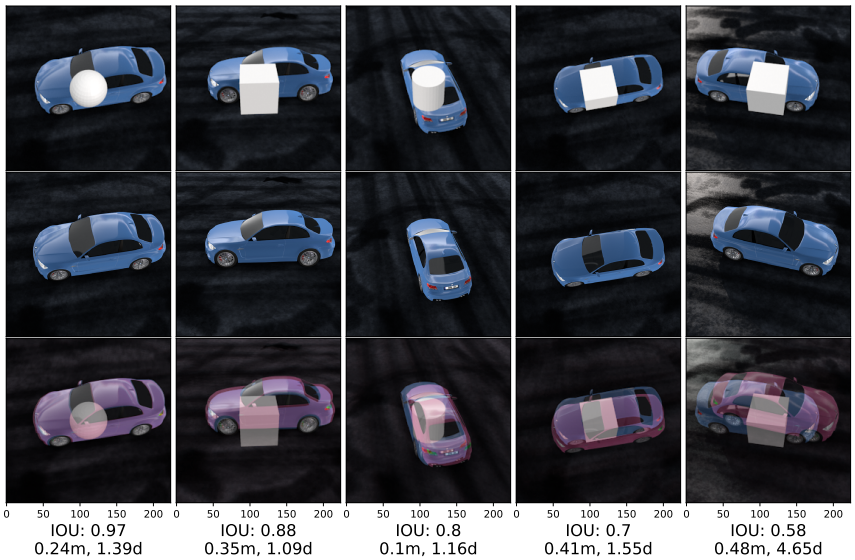

Figure 3.7: Archetypal representations for each IOU threshold.

Table 3.5 also shows that how well model performed on the task and the number of worst case scenarios across different models. A remarkable number is the estimated angles are much closer to the ground truth labels. We believe this result is due to the application of a quaternion constraint in Equ.(3.6). We have also observed that application of such constraint on both models improves the average IoU by decreasing the number of worst case predictions despite it has a negligible detrimental effect on high accuracy predictions.

Following the tradition, we have also tested our models performance on *Cambridge Landmark dataset* and *7 Scenes dataset*. In Table 3.2, we demonstrate that our model performs comparably to the state of the art camera relocalization using other deep neural networks while having one tenth of trainable parameters.

We have also tested our model's generalization capabilities when the training is done with views obtained from a closeby camera and test the performance on images obtained further away from the object. For this, we have generated 1000 test images rather distant from the target object. Camera radii r_1 and r_2 are selected as 15 and 30 meters for this test dataset. Training images were obtained within 15 meters. Table 3.3

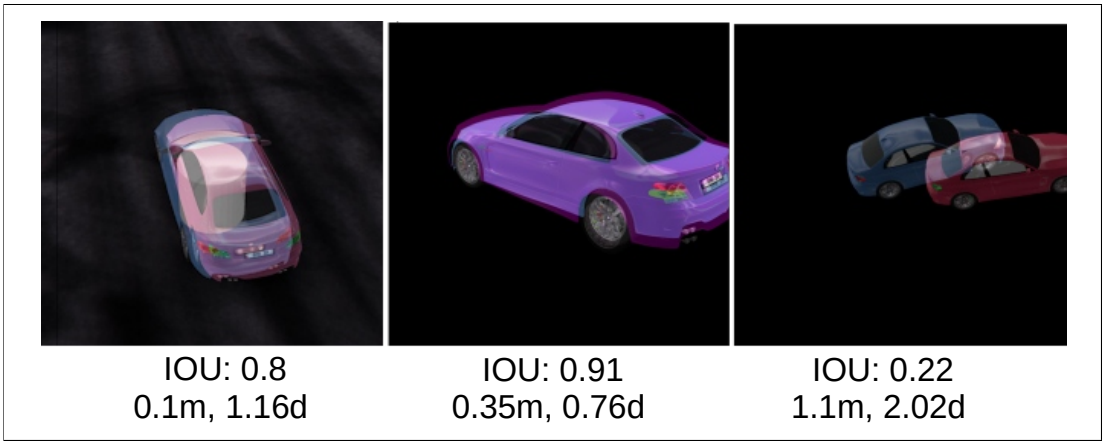

Figure 3.8: Performance of proposed model on out of distribution synthetic data.

Table 3.3: The model performance tested on images with larger ranges of motion.

Motion Range	IoU Score	Location Error	Angle Error
5-15m (Similar to Training Data)	0.845	0.21	1.07°
15-20m	0.842	0.22	1.07°
20-25m	0.46	0.54	4.64°
25-30m	Failed	Failed	Failed

shows that our model performs well up to 20 meter radius and breaks down after that. This is expected as object detils become indiscernable after certain distances.

Another robustness test is conducted to calcutale orientation variations. In the training data we let only portion of orientations present for each car type (see Figure 3.9). We fix the tilt angle w.r.t. the object while varying the pan angle freely. We make sure that the entire range of orientation is covered by the two different cars. When the first car covers the angle range $0^o - 45^o$, the second car covers the next range $30^o - 60^o$ and so on. The trained model is then tested on a car that was imaged for the remaining range of pan angles (not used in the training data for that car). As the results show in Table 3.4, the model successfuly learns the representation of the car for non-existing poses. This suggests that pose transfer from one car to the other is accomplished by the model. Of course, we can not conclude that the pose of a completely new can be estimated by the model as good as the existing cars that is used in test. See Figure 3.8.

Figure 3.11 shows the effect of the choice of regularization term on the validation

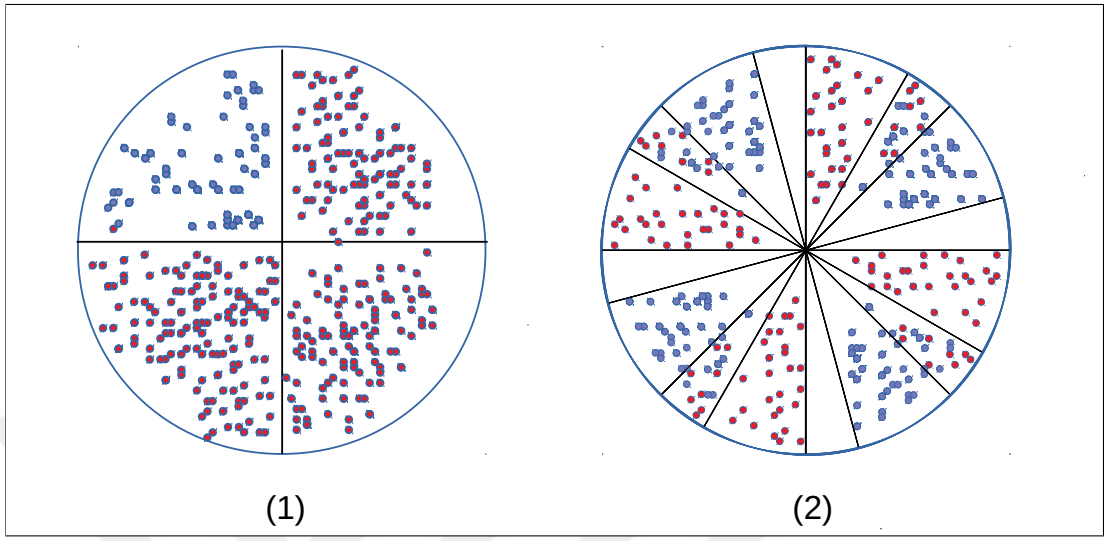

Figure 3.9: 2D representation of camera pose distribution for obtaining training data.

Figure 3.10: Visualization of attention maps.

Table 3.4: The model performance on images with diverse orientation ranges.

Motion Range	IoU Score	Location Error	Angle Error
5-15m (Similar to Training Data)	0.845	0.21	1.07
Sparse data	0.812	0.45	1.64°

performance during training. As discussed earlier, we expect $\hat{Q}$ to behave better than the direct loss or the quadratic regularization $(Q-1)^2$. Even though exponential regularization starts slowly, it picks up and gives slightly better validation error. The positive effect of this regularization on other models can also be seen in Table 3.5. When we apply this regularization on PoseNet, although higher accuracy levels did not change, for the lower accuracy levels, performance has seen a dramatic increase (columns under < 0.2 and < 0.1). We also used this regularizer on another model. When applied in training PoseNet, the number of iterations to get to the same performance are decreased by about 50% (see Figure 3.11).

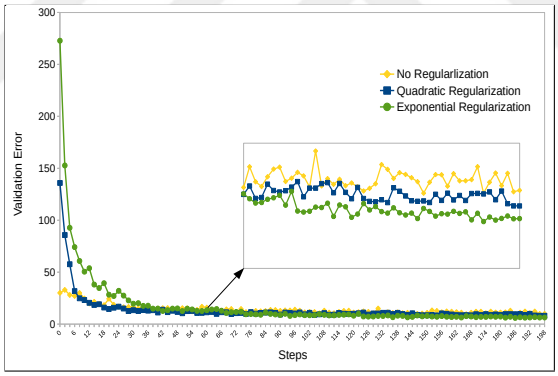

Figure 3.11: Comparison of different regularization functions on training and validation loss.

Our model shows good experimental perforance on synthetic as well as real data. As a way to get an insight into the model we have looked into where on the object the model is given attention. Figure 3.10 shows example saliency maps produced by our model. Originally proposed in[Simonyan et al., 2013], saliency is taken as the magnitude of the gradient of the loss function with respect to the pixel intensities. In other words, it indicates how a region of the image contributes to the pose estimation of the model. We constructed these figures by adding the gradient of the given image on

Table 3.5: Comparision of Accuracy in selected models.

Model	Param	Med. IOU	Accuracy				
			>0.9	>0.8	>0.7	<0.2	<0.1
PoseNe	22M	0.547	47	269	426	202	157
PoseNet2(Const.)	22M	0.675	53	297	586	57	44
PoseNet3(Const.,Geo.)	22M	0.857	363	792	908	35	28
Our Model(Const.)	2.4M	0.739	139	508	722	36	30
Our Model(Const., Pro.)	2.4M	0.845	345	790	912	15	12

top of the test images. It is noticeable that model is learning strong visual cues around the edges and contours. Contrary to classical SIFT approach, our model also attended over textureless patches in some cases.

3.4.5. Summary

This chapter, A real-time monocular camera pose estimation algorithm is presented. Proposed model is a small convolutional neural network that is trained to directly estimated 6-DOF camera pose from an RGB image. Our model is designed to run on real-time devices with low memory and computation power. Additionally, we demonstrated that an an algebraic constraint on rotation matrices could enable faster convergence and yields better results.

4. OBJECT ASPECT CLASSIFICATION AND 6-DOF POSE ESTIMATION

Recent works focus on a multi-stage approach, which first detects key-points followed by perspective -n-point pose estimation algorithm and a pose refinement procedure. In this chapter, we show that adding a classifier block estimating the predefined aspects of the objects improves the multi-stage process. This is due to the fact that the additional classifier acts as a constraint simplifying the required neural network and at the same time yielding better key-point selection. We reduce the search space for the key-point selection and exclude false-positives by mapping the appearance of an object to an aspect. The simplified neural network allows faster inference and a smaller footprint. Our experiments show that our hypothesis performs similar to the state-of-the-art on three different datasets. We also show that an off-the-shelf refinement process can further improve our results to surpass state-of-the-art on several objects. Another advantage is, the proposed pipeline can run efficiently on real-time due to the smaller neural network backbone used.

4.1. Introduction

Object pose estimation recovers the orientation and translation of an object within a given view with respect to a reference coordinate frame. It is an essential part of a successful application in augmented reality, robotics and autonomous vehicles. For instance, in an augmented reality setting, a user's interactions with an object are heavily depended on its pose. If the object pose is incorrect, any augmentation will look unnatural. In robotics, a robot must estimate the pose of the targets so that it can perform tasks such as grasping and relocating. Pose inference from a single RGB image can be very challenging. Many of the challenges come from the scene itself, such as background clutter, varying lighting and imaging conditions. Other challenges are due to the object itself, like self-occlusion, lack of texture or texture changes due to wear and tear.

Traditionally, pose estimation problems are solved by corresponding low-level features between the image and object. These features are usually hand-crafted based on edges, corners or small contours [Lepetit and Fua, 2005]. They do not include global information (such as object topology) or the semantics of the scene. Furthermore, these techniques do not cope with textureless objects well. Since underlying feature selection algorithms [Lowe, 2004; Rublee et al., 2011] usually require rich textures on the object to build robust features. Finally, these solutions perform poorly under challenging set-ups like occlusion or scene clutter. Recently deep-learning researchers have proposed a broad range of algorithms [Gupta et al., 2019; Peng et al., 2019; Abadi et al., 2016; Persson and Nordberg, 2018; Li et al., 2020] to tackle the pose estimation problem. These methods seem to capture object representations even when there is a lack of texture. They can learn object semantics (related to aspects or viewpoints of the object) through a significant amount of training. Some of these studies, either directly estimate the object pose using a classification or regression scheme. And the rest use deep pre-trained convolutional networks (CNN) as high-quality semantic feature selectors and apply a robust (e.g., random sample consensus algorithm - RANSAC) pose estimation algorithm (e.g., 3D to 2D perspective n-point PnP).

When analysed in depth, these methods look like they are memorizing aspects corresponding to a variety of poses of the object. Provided that one has many such aspects in the training data, a large enough network can learn a global semantic representation for features to be used in the PnP algorithm. Following this observation, instead of learning these aspects implicitly, we propose a new estimation pipeline based on the appearance and key-points related to the geometry of the object. The CNN part of the network has two purposes: 1) capturing the semantic feature representation, and 2) classifying a set of aspects explicitly. The first follows the existing methods while the second helps to reduce the requirement for memorization leading a lighter CNN with fewer parameters and comparable if not better performance.

The proposed architecture runs at 54 fps on a relatively low power GTX 1070 GPU along with an i5 processor. To accomplish fast and accurate estimations, we construct a

fixed number of view-point maps which we reduce to aspect structures. Following aspect construction, we select key-points based on the silhouette and the low-level invariant features represented as dense vectors on the target object surface. Finally, we estimate the pose with a traditional PnP. Our aspect pipeline allows us to select an appropriate set of key-points in the aspect graph, thus reducing the computational power need and increase the robustness of our feature hypothesis. We demonstrate the performance of our pipeline and the effects of the aspect proposal on highly challenging LINEMOD, Occlusion LINEMOD, and T-LESS datasets. Our results show that the proposed architecture is comparable to the state-of-the-art, albeit with a simpler architecture and training procedure. Moreover, we show that by utilizing an ad hoc pose refinement stage, our method can beat the state-of-the-art methods by a significant margin.

We summarize our contributions as:

- We introduce a simple aspect class representation for compact 3D objects. This representation is used to label the existing views of the objects as aspects which in turn is modeled to estimate the aspect of an object in a given image. Aspect predicting leads to high confidence key-point vectors hence more accurate pose estimation results.
- The proposed pipeline is significantly faster and comparable in accuracy to the state-of-the-art pose estimation algorithms. We demonstrate this is the result of backbone architecture and reduction in the size of key-point space during PnP+RANSAC.
- We also demonstrate a significant improvement in state-of-the-art pose estimation results by just incorporating an off-the-shelf refinement stage.

The rest of the manuscript is organized as follows. Section 4.2 gives a summary of the contemporary and traditional solutions to the 6DoF pose estimation problem. In section 4.3, we present our method using aspect and key-point construction. In section 4.5, we present implementation details, training procedure, experimental setup and estimation pipeline. Section 4.7 details the datasets, evaluation metrics, results and

comparisons. Finally, we discuss the failure cases, future directions and portability of the proposed solution in conclusion.

4.2. Related Work

Traditional methods can roughly be split into two categories. Methods int the first category infer the object pose using local features extracted from the target image and matches to the 3D model. Using these 2D-3D correspondences 6DoF pose can be recovered [Zhu et al., 2014; Martinez et al., 2010]. These methods usually show high robustness against partial occlusion. However, they require robust visual features to operate and fail when the objects are textureless.

Contemporary pose estimation approaches heavily employ deep learning methods either by directly estimating the pose or use it as part of a multi-stage pipeline [Akgul et al., 2016]. In the seminal work of Xiang [Xiang et al., 2018], PoseCNN uses a CNN architecture to estimate 6DoF pose by regressing to a quaternion representation. However, their method heavily suffers when the object is occluded. Moreover, Obwerger et al [Oberweger et al., 2018] show that occluders have a corrupting effect on CNN activations far beyond the receptive field. In response to those failures, many researchers adopt a multi-stage pipeline similar to the traditional methods. In the first stage, they predict the 2D key-points and then compute the pose with PnP algorithm. These methods train CNNs to detect semantic key-points. Lepetit et al [Rad and Lepetit, 2017] uses segmentation and sparse correspondences for detecting the 3D bounding box. Tekin et al [Tekin et al., 2018] use YOLO [Choi et al., 2019] architecture to estimate object key-points on a multi-resolution feature map. In a dense-approach, contrary to generic sparse approach, every object pixel has a contribution to the pose estimation. Doumanoglou [Doumanoglou et al., 2016] and Kehl [Kehl et al., 2016] use CNN to sample RGB-D image patches and extract features for the voting. Similar to our work Peng et al [Peng et al., 2019] uses the Farthest Point Sampling algorithm (FPS) to select key-points on the target object. Then they use a heavy-weight pre-trained CNN [He

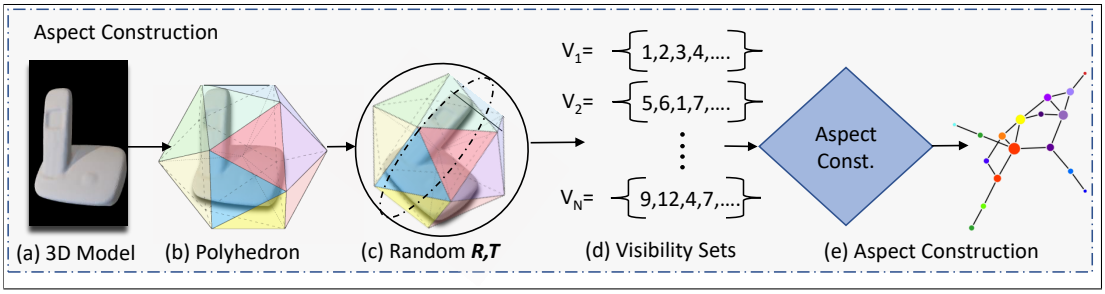

Figure 4.1: Illustration of construction of pseudo aspect graph for a given object.

et al., 2016] to predict the key-points and a segmentation map. Furthermore, they notice that the quality of predicted key-points may vary. Thus they introduced *Uncertainty-driven PnP* that minimizes the Mahalanobis distance between each predicted spatial key-point distribution and the 3D correspondence.

Learning-based methods achieve better performance than traditional methods, largely due to the ability to learn powerful feature representations for pose estimation. Many studies include a refinement stage to improve the overall accuracy of the predictions. DeepIM [Li et al., 2020] iteratively refines the pose by matching the rendered images against the observed images and minimizes a similarity measure. DPOD [Zakharov et al., 2019] combines initial pose estimation process with refinement stage to a single end-to-end trainable network with ADD metric serving as a refinement loss.

4.3. Aspect-driven 6DoF Pose Estimation

In this work, we propose a novel pipeline for 6-DoF object pose estimation (See Figure 4.2). Given an RGB image, our method estimates a transformation $(R;t)$ where R represents the rotation and t represents the translation of the object with respect to a reference coordinate frame. Similar to [Peng et al., 2019; Gupta et al., 2019], our pipeline estimates fine-grained object pose in multiple stages. At first, we predict the *aspect* of the target object based on the appearance by using a CNN. On a parallel stream, we predict semantic key-points using a fast pre-trained CNN [Choi et al., 2019]. In the final stage, we select K key-points related to the aspect and compute the object pose using an off-the-shelf PnP solver.

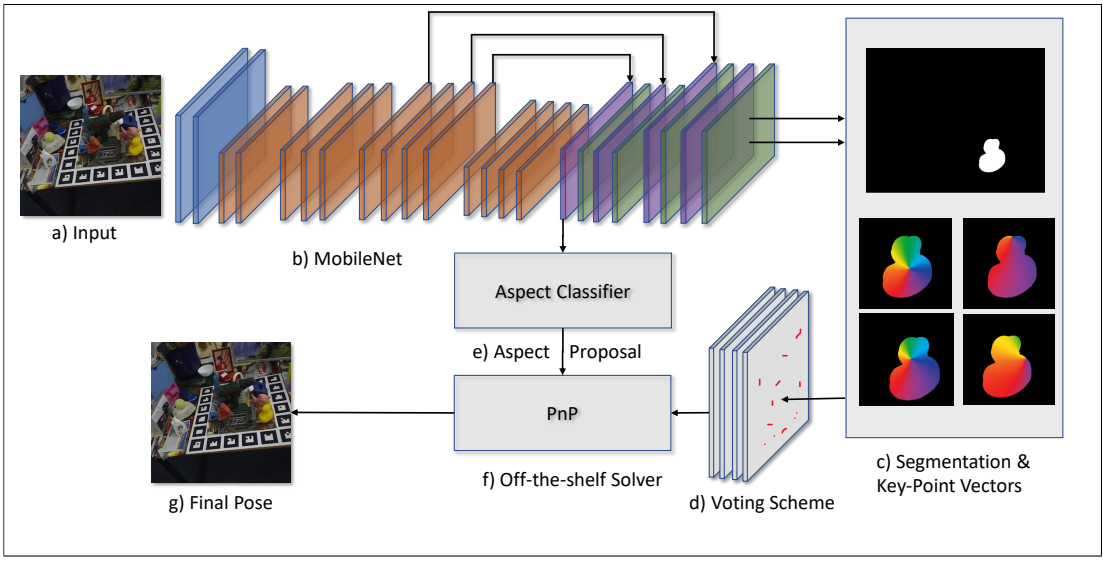

Figure 4.2: Proposed pose inference pipeline.

4.4. Object aspect prediction

In computer vision, aspect graphs have been heavily studied in the early 1990's. Researchers constructed algorithms for automatically computing the aspect graphs of polyhedra, general curved objects and even objects with articulated connections between parts. However, much of the work in this area has a somewhat theoretical flavor and did not find a practical application in the mainstream research [Faugeras et al., 1992]. True aspect-graph construction is very difficult and not very useful in pose estimation due to complexity of the construction algorithms and resulting graphs being very large. We, therefore, propose to construct a pseudo aspect graph for pose estimation purposes.

The pseudo-aspect construction algorithm (see Fig. 4.1) determines the final aspect of the 3D object represented as a polyhedron $F = \{F_1, F_2, F_3 ... F_n\}$ where F_i denotes a face of the regular polyhedron. We calculate the projected areas of the visible faces on to the camera plane. The face with the maximum projected area is selected to be the *anchor face*. We calculate the visibility of remaining faces by determining the ratios of projected areas of each face to the *anchor face*. If the value is below a certain threshold θ, we assume that that face is invisible. From the visible faces, we construct a visibility set that includes all the visible faces from a given viewpoint(Object pose in this context). Finally, we cluster the visibility sets according to their size and their

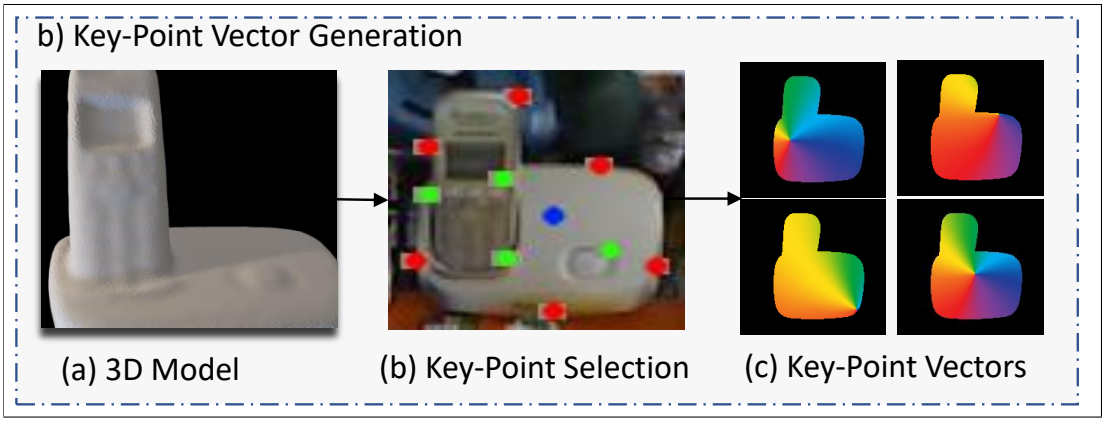

Figure 4.3: Illustration of key-point selection

contents using an algorithm detailed in 4.5.1.

The rationale behind using the pseudo aspect construction algorithm is that a traditional aspect-graph approach tracks visual cues like edges or corners. Thus graph construction is based on the visibility of local features. However, this approach would be infeasible in our method. Since given a set of features, the size of a vertex-graph is usually quadratic with respect to the feature size resulting in unmanageable number of aspects. Although it is possible to prune and cluster the visibility-set with graph algorithms, we aim to obtain a fast, small scale data generation and estimation pipeline. Our method allows us to control the size of the aspect-space by using the type of regular polyhedron and the threshold parameter θ.

4.4.1. Key-point selection

Recent studies [Xiang et al., 2018; Tekin et al., 2018] use corners of the bounding box of the 3D object as key-points. However, detailed analysis in [Peng et al., 2019] demonstrates that off-surface key-points results in larger localization errors in many cases. Curious readers may refer to the manuscript to examine the comparison and ablation studies. Following the spirit of [Peng et al., 2019], we select a set of key-points on the object. Instead of directly using farthest point sampling (FPS), we select the key-points to be uniformly spread in the given aspect both in radial and angular coordinates. Along with appearance-based selection, uniform placement of the key-points in the respective aspect helps extract locally better features. Even though the object may lack

texture, the image of the current aspect may still have distinctive textural appearance due to shading and self-occlusions. SIFT-like 2D feature extractors [Hess, 2010] can capture these within the given aspect enriching the representation (or the semantic) of the 3D key-points.

Our key-point selection algorithm goes as follows. First, we select the center of the bounding box of the object as the first key-point. We emit N_1 lines from the object center which are separated by a pre-defined angle α. Then, we take the intersection points between the emanating lines and the borders of the object as these N_1 key-points. Another set of key-points with cardinality N_2 is selected using a more traditional method [Lowe, 2004]. In this step, we extract invariant features with an off-the-shelf feature extractor such as SIFT. Extracted features on the object are used as key-point candidates. Finally, using the key-points generated in the first step, we select a candidate that is farthest to the current key-points until the set is complete. In contrast to prior work, our multi-step, multi-faceted key-point selection strategy enforces the network to pay attention to both local invariant features (aspect dependent) and the global structure and the appearance of the object. An illustration of the method can be seen in Figure 4.3.b.

Our key-point localization technique is very similar to *Peng et al* [Peng et al., 2019]. For each target image, our pipeline predicts a dense-vector $v_i(p)$ with $(N_1{+}N_2 + 1)$ normalized 2D vectors for each pixel. These unit vectors represent the 2D direction of the key-point relative to the pixel p. During inference, we calculate the location of the key-point by using the following algorithm. First, we choose N random pixels from the estimated dense-vectors and take intersections of their prediction vectors. If calculated intersections are within a close range ν, our pipeline marks this hypothesis as a valid key-point estimation. Else, we select all pixels and vote for the intersections and most confident is selected as a valid key-point estimation. More specifically, we calculate the confidence score of a key-point $C(p)$ as

$$C(p) = \frac{\delta(p_1)}{\delta(p_1) + \delta(p_2)} \tag{4.1}$$

where p_1 is the highest voted key-point and the p_2 second highest voted key-point. Finally, δ is the counting function that shows the number of votes each point received. We omit key-points that have lower confidence scores than 0.95.

4.4.2. Pose estimation

We estimate the final pose by using E-PnP[Lepetit et al., 2009] on high confidence key-points supported by voting mechanism. We use the estimated aspect for selecting the associated key-points with the aspect. This choice directly reduces the search space of PnP-RANSAC. Another advantage of the approach is on the key-point estimation architecture. We reduce the number of key-points to be estimated for a given view. This choice translates to a light-weight neural architecture and faster inference.

We have also tested our key-points by using the method proposed in [Peng et al., 2019].

4.5. Implementation

We use TensorFlow2 [Abadi et al., 2016] with Python bindings for dense key-point estimation. For the PnP solver, we use Lambda-Twist [Persson and Nordberg, 2018]. We detect SIFT features by using OpenSIFT [Hess, 2010]. We use Blender for rendering additional images and computing projection matrices and labels. We perform all our experiments on an RTX 2060 Super GPU with 32 GB of RAM and a GTX 1070 with 16 GB of RAM. We use the backbone architecture used in MobileNet [Sandler et al., 2018] with a few custom changes similar to PvNET. First, when the feature map of the network is reduced to the size $H/8 \times W/8$, we do not downsample the feature map anymore. Secondly, we replaced subsequent convolutions with dilated convolutions [Yu and Koltun, 2016]. Then, we repeatedly perform skip-connection, convolution and upsampling on the feature map, until its size reaches $H \times W$. Finally, we added an $H \times W \times 2 \times (N_1 + N_2 + 1)$ output tensor for dense key-point estimation. In order to choose correct pixels for key-point vectors, we place an additional semantic

segmentation predictor with a size of $H \times W$ (See Figure 4.2). Aspect classification is performed on a parallel stream with soft-max cross-entropy based output at the end of the first *convolution* block of the decoder.

4.5.1. Aspect construction

Given a 3D model, we construct the smallest encompassing *icosahedron*. Following the strategy in 4.4, we select visibility threshold θ as 0.2. In other words, if the ratio of the visible area of a face to the anchor area is less than 0.2 we assume that the face is invisible. In addition to its simplicity, employing an area-based ratio can allow the usage of any regular polyhedra in different setups. We visualized these stages in Figure 4.1.a. At first, we construct the aspect graph using [Plantinga et al., 1986] from the visibility sets. However, this method was inefficient since it produced aspect-graphs larger than the size of the average visibility set ($O(n^2)$ on average). Thus, we change aspect graphs using a simple recursive algorithm based on the similarities of the visibility-sets:

In this algorithm, the similarity is measured between the intersection of two sets. If the number of shared elements is at least %80 of the smaller set then two sets are count as similar. Hyper-parameter ω defines the number of classes (i.e: distinct aspects). We choose ω as 64 in all our experiments based on an exploratory analysis on a few selected models.

We use soft-max with cross-entropy loss on aspect classification and regular L_2 loss with sigmoid activation on aspect confidence scores. The total aspect loss is:

$$L_{asp} = L_{cls} + L_{conf} \tag{4.2}$$

where $L_{asp}, L_{cls}, L_{conf}$ define aspect loss, classification loss and confidence loss. Respective losses are formulated as:

$$L_{cls} = \frac{1}{N} \sum_{i=1}^{N} -\hat{y}_i \log(y_i) \tag{4.3}$$

$$L_{conf} = \frac{1}{N}\sum_{i=1}^{N} \quad \|\hat{C}_i - C_i\|_2 \tag{4.4}$$

where y and $\hat{y}$ defines the predicted and ground truth aspect class labels and C_i and $\hat{C}_i$ defines the confidence score for the rest of the aspects.

4.5.2. Key-point training

Given an RGB image, we train our network by predicting the dense direction vectors $v_i(p)$ in 4.4.1. Let $v_i(p)$ and $\hat{v}_i(p)$ be a predicted key-point vector for pixel p and the ground truth vector respectively. We apply a smooth L_1 loss following [Girshick, 2015].

$$L_p = \sum_{n=1}^{N}\sum_{p} \quad (v_i(p_x) - \hat{v}_i(p_x)) + (v_i(p_y) - \hat{v}_i(p_y)). \tag{4.5}$$

Finally, combining equations 4.2 - 4.5

$$\begin{aligned} \mathscr{L}_{total} = \sum_{p} \quad & (v_i(p_x) - \hat{v}_i(p_x)) + (v_i(p_y) - \hat{v}_i(p_y)) + \\ & \frac{1}{N}\sum_{i=1}^{N} -\hat{y}_i \log(y_i) + \frac{1}{N}\sum_{i=1}^{N} \quad \|\hat{C}_i - C_i\|_2 \end{aligned} \tag{4.6}$$

gives us a single objective function.

We use a pre-trained MobileNet [Sandler et al., 2018] implementation in TensorFlow2 [Abadi et al., 2016]. Our model is optimized using Adam [Kingma and Ba, 2015] for 100 epochs. The initial batch size is 8. We double the batch size in 40th and 80th epochs for stability. We augment the training data slightly by randomly perturbing hue, saturation and exposure. Furthermore, for occlusion, we place a random geometric-shape (circles, rectangles and triangles) with random color on top of the synthetic images. We do not use any further data augmentation. The hyper-parameter angle α value is selected to be 60° resulting in 7 points with the object center for N_1 and the number of SIFT features, N_2, is set to 6. Thus the total number of key-points is set to 13 in all our experiments.

	Object														
Method	1	2	3	4	5	6	7	8	9	10	11	12	13	14	19
Vidal-18	43	46	68	65	69	71	76	76	92	69	68	84	55	47	57
Drost	53	44	61	67	71	73	75	89	92	72	64	81	53	46	59
Hodan	66	67	72	72	61	60	52	61	86	72	56	55	54	21	55
Ours	51	53	64	71	61	70	60	59	59	68	81	84	70	53	71
Ours+DeepIm	56	56	64	72	67	74	61	62	61	77	81	84	71	53	72

Table 4.1: Accuracy is calculated using *VSD* metric as defined in [Hodan et al., 2018]. DeepIM is integrated as a post-refinement stage. Results demonstrate that refinement stage favors some objects(1,5,10) than the others.

.

Table 4.2: Comparison of 2D projection accuracy on LINEMOD dataset. Refinement column denotes an additional post refinement stage for the pipeline.

	Method w/o ref.					Method w/ref.	
Object	BB8	Tekin	PvNET	CullNet	Ours	BB8	DeepIM
ape	95.3	92.10	99.23	97.7	99.0	96.6	98.4
bench	80.0	95.06	99.81	99.0	99.0	90.1	97.0
cam	80.9	93.24	99.21	97.9	96.2	86.0	98.9
can	84.1	97.44	99.90	98.9	99.1	91.2	99.7
cat	97.0	97.41	99.30	98.7	99.2	98.8	98.7
driller	74.1	79.41	96.92	96.4	95.5	80.9	96.1
duck	81.2	94.65	98.02	97.0	95.4	92.2	98.5
eggbox	87.9	90.33	99.34	98.7	98.4	91.0	96.2
glue	89.0	96.53	98.45	98.2	98.2	92.3	98.9
hole	90.5	92.86	100.0	99.0	99.3	95.3	96.3
iron	78.9	82.94	99.18	97.2	96.9	84.8	97.2
lamp	74.4	76.87	98.27	95.4	94.9	75.8	94.2
phone	77.6	86.07	99.42	95.6	98.0	85.3	97.7
average	83.9	90.37	99.00	89.3	97.5	97.7	97.3

4.5.3. Pose inference

For inference, we take the estimated aspect and select the key-points from the aspect as canonical representation. Then we estimate the object pose by using the canonical and estimated key-points using LambdaTwist PnP algorithm. Furthermore, we show that, with some tweaks, the accuracy of the proposed pipeline can further improve by applying post-refinement stages [Gupta et al., 2019; Li et al., 2020; Zakharov et al., 2019].

In this section, we present the results of our experiments and show comparison with the state-of-the-art based pose estimators using RGB images.

4.5.4. Datasets

We train our network on well-known LINEMOD [Hinterstoisser et al., 2013] and T-LESS [Hodaň et al., 2017] datasets. We have also tested our hypothesis on Occlusion LINEMOD [E. Hinterstoisser et al., 2014] dataset. In addition to the training samples given in the dataset, for each of the 13 objects, we also generate 10000 images whose object poses are sampled uniformly based on the distribution of the corresponding dataset. These images are synthesised on the random images from PASCAL-VOC dataset [Everingham et al., 2010] using the approach described in [Dwibedi et al., 2017]. LINEMOD is one of the standard benchmark datasets for 6DoF pose estimation. The dataset contains 13 object classes with around 1200 images per class. This dataset is highly challenging due to background clutter, variations in illumination and texture-less objects. Following the literature, we take 200 images from the original dataset for training and the remaining are used in testing. Occulison LINEMOD is a variation of the original LINEMOD dataset. 8 objects in the LINEMOD dataset are additionally annotated. In total 1215 frames with heavy occlusion are selected from the original dataset. We do not perform any training using occlusion LINEMOD dataset. We use this dataset for testing. T-LESS is a recently built dataset which contains around 30 objects with 1296 images per class. In our experiments, we do not use the whole T-LESS dataset. We take 15 objects and follow a data generation procedure similar to the LINEMOD.

4.6. Evaluation metrics

We follow the literature to evaluate the performance of 6DoF algorithms.

- 2D Projection: The 2D Projection metric measures how close the 2D projected vertices are to the ground-truth, in pixel domain. Following the common practice in the recent papers, we consider the pose as correct if the mean 2D projection error is below 5 *pixels*.
- ADD Metric: Average Distance (ADD) computes the mean of the pairwise

distances between the ground truth pose and the estimated pose:

$$ADD = \frac{1}{N} \sum_{x \in M} \| (\hat{R}x + \hat{t}) - (Rx + t) \|_2 \tag{4.7}$$

where $\hat{R}$,$\hat{t}$ are predicted rotation and translation R,t are ground truth labels. M describes the set of model points, N defines the size of the set. The 6-DOF pose is considered correct if the ADD score is smaller than a threshold. This threshold is not a constant value and depends on the size of the target object. Following the previous works, we set this threshold to be 10 percent of the target objects diameter. For symmetric objects, the matching between points can be ambiguous for some views. In those cases, we calculate the accuracy by ADD(S) [Hinterstoisser et al., 2013].

- Visible Surface Discrepancy (VSD) : We additionally use the metric provided by [Hodan et al., 2018] available online (see the paper for the link) as an alternative method of measuring pose accuracy for occluded objects. This metric evaluates the pose accuracy taking into account the amount of occlusion the object has in the test image. After rendering the synthetic image with the given pose, VSD rewards the foreground pixels with small errors in predicted distances (or errors in dense 2D projections).

4.7. Experiments

We conduct experiments to compare the proposed method to the state-of-the-art, evaluating the contribution of aspect classification and utility of the pipeline as a backbone of a refinement process.

4.7.1. Results

We present our results on T-LESS dataset in Table 4.1. We compare our results to the ones presented in [Hodan et al., 2018]. However, this dataset includes many symmetric objects. In those cases, we construct aspect graph on half of the view-space.

Table 4.3: Comparison of ADD(S) accuracies on LINEMOD dataset. Right-most four columns shows the inclusion of a post-refinement stage.

	Methods w/o refinement.						Methods w/ refinement.			
Object	BB8	SSD	Tekin	PvNET	CullNet	Ours	BB8	DPOD	DeepIM	Ours*
ape	27.9	2.6	21.6	43.62	55.1	82.2	40.4	87.7	77.0	86.9
bench	97.5	15.1	81.8	99.90	89.0	94.0	98.45	91.8	97.5	94.7
cam	40.1	6.1	36.6	86.86	66.2	95.4	96.07	55.7	93.5	96.1
can	48.1	27.3	68.8	95.47	89.2	96.4	99.71	64.1	96.5	97.0
cat	45.2	9.3	41.8	79.34	62.6	94.7	96.1	75.3	82.1	96.3
driller	58.6	12.0	63.5	96.43	88.6	89.7	98.80	74.4	95.0	94.2
duck	32.8	1.3	27.2	52.58	44.3	86.3	75.8	41.8	77.7	86.3
eggbox	40.0	2.8	69.6	99.15	97.1	98.3	57.8	99.9	97.1	97.8
glue	27.0	3.4	80.0	95.66	94.6	98.1	41.2	96.8	99.4	95.1
hole	42.4	3.1	42.6	81.92	68.9	84.9	67.2	86.9	52.8	84.9
iron	67.0	14.6	74.9	98.88	90.9	98.9	84.7	100	83.0	98.9
lamp	39.9	11.4	71.1	99.3	76.5	96.8	94.2	94.0	76.9	97.2
phone	35.2	9.7	47.73	92.41	54.0	94.7	67.6	93.0	86.1	95.0
average	43.6	9.1	55.9	86.27	78.3	92.1	62.7	95.15	90.37	93.9

Table 4.4: Comparison of ADD(S) metric accuracies on different threshold values Occlusion LINEMOD

Method	Ours w/o aspect			Ours			Ours+DeepIM		
threshold	0.05	0.10	0.15	0.05	0.10	0.15	0.05	0.10	0.15
ape	0.2	16.6	47.1	14.3	28.3	63.0	22.1	34.7	64.2
can	6.3	64.2	65.9	48.5	70.4	73.1	55.1	71.6	75.7
cat	0.2	17.1	20.2	4.2	28.1	47.7	8.1	35.4	49.1
driller	4.0	24.3	26.6	32.6	58.3	71.6	44.8	58.9	71.6
duck	1.1	64.1	69.2	44.7	69.0	77.4	51.3	70.4	79.7
eggbox	7.0	41.0	48.5	46.1	57.0	72.8	49.2	65.1	73.2
glue	10.0	50.9	53.7	35.4	54.2	68.1	39.2	59.1	68.3
hole	16.6	39.6	41.3	33.1	50.7	66.4	41.7	56.8	67.5
average	5.6	39.9	46.5	28.6	51.9	67.5	38.9	56.9	68.6

We present the accuracy measure as VSD. We show the accuracy on 2D projection metric on LINEMOD dataset in Table 4.2. Our method performs similar to PvNET since both methods share a common spirit despite having different neural network backbones and key-point selection algorithms. As the state-of-art methods report similar performances, a closer analysis is needed. We present a detailed comparison of our strategy on different error thresholds in Fig. 4.4. In addition, Table 4.3 demonstrates our results based on ADD(S) metric. Proposed pipeline surpasses the state-of-the-art without any refinement stage. As a result, we present Tables 4.3 and 4.2 as a confirmation that the method performs comparable to the state-of-the-art.

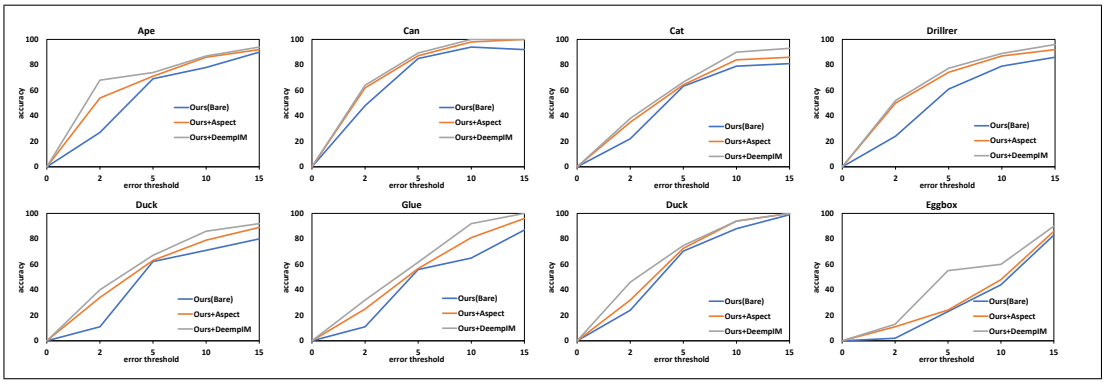

Figure 4.4: Accuracy of proposed method on different object classes in Occlusion LINEMOD dataset.

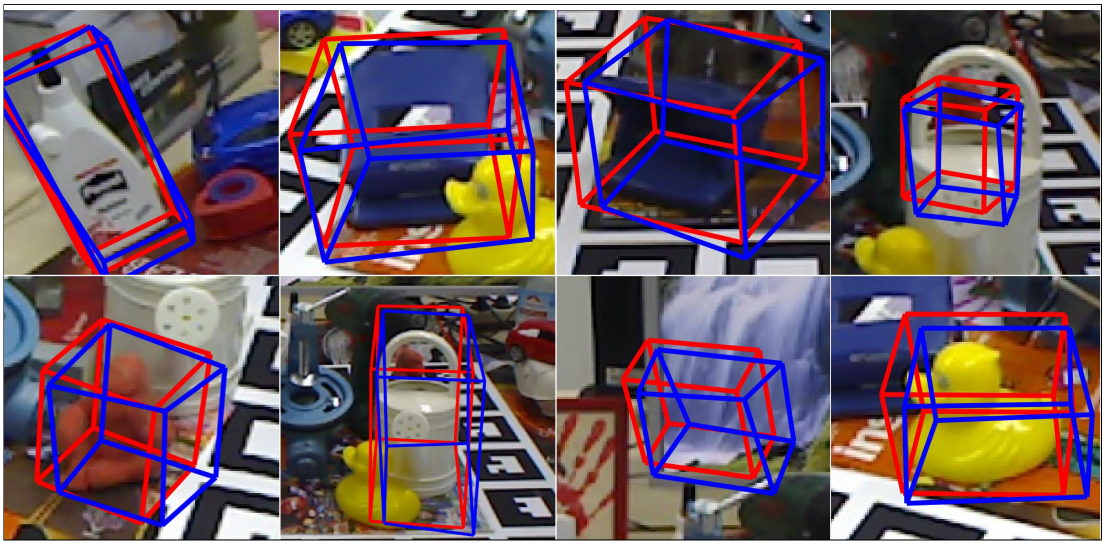

Figure 4.5: Qualitative results on refinement stage on Occlusion LINEMOD dataset.

Table 4.5: Comparison of ADD(S) metric accuracies on Occlusion dataset.

	Methods w/o refinement				Methods w refinement			
Object	Tekin	PoseCNN	PvNET	CullNet	Brynte	DeepIM	Ours	Ours*
ape	2.48	9.6	15.81	21.97	40.85	59.18	28.3	34.7
can	17.48	45.2	63.30	24.52	82.44	63.52	70.4	71.6
cat	0.67	0.93	16.68	9.77	35.64	26.24	28.1	35.4
driller	7.66	41.4	25.24	23.62	71.33	55.58	58.3	58.9
duck	1.14	19.6	65.65	26.11	49.08	52.41	69.0	70.4
eggbox	-	22	50.17	20.43	57.28	62.95	57.0	65.1
glue	10.08	38.5	49.62	28.02	62.90	71.66	54.2	59.1
hole	5.45	22.1	39.67	41.4	43.14	52.48	50.7	56.8
average	6.42	24.9	40.77	24.48	55.33	55.50	51.9	56.9

Table 4.6: We deduce that high number of view-points clearly benefits proposed pipeline objects with normalized shapes(ape)

	ape	bench	cam	can	cat	drill	duck	egg	glue	hole	iron	lamp	phone
Ours(Orig)	82.2	94.0	95.4	96.4	96.1	89.7	75.8	98.3	98.1	84.9	98.9	94.0	93.0
Ours(n=16)	71.4	92.0	93.2	96.3	94.1	83.1	66.4	98.1	97.3	76.5	94.2	94.0	90.9
Ours(n=32)	80.8	94.1	95.4	96.0	96.0	87.2	75.8	98.1	97.9	84.0	97.4	94.0	93.1
Ours(n=96)	80.4	94.7	96.2	96.8	95.4	89.7	75.8	98.3	98.1	85.1	98.8	94.0	92.0

4.8. Ablation studies on Occlusion dataset

In Table 4.5 we compare our results to the state-of-the-art on Occlusion LINEMOD dataset. Even without the refinement stage, our method performs comparably to the state-of-the-art. Furthermore, with the inclusion of DeepIM to our pipeline, we observe a significant (5 point) rise in performance. However, our pipeline still carries the weakness against ape and cat objects. When we examined the particular failure in those objects, we observe that they have larger occluded area than the rest of the objects. Even with the naked eye, we have a hard time recognizing these objects in a crowded scene.

Table 4.4 shows the performance of our method with and without the aspect graph and refinement stages on Occlusion LINEMOD dataset. Note that without these stages, our algorithm is very similar to [Peng et al., 2019] except that our network is much lighter. We show the performance of these three methods using the ADD(S) measure. Moreover, following the DeepIM [Li et al., 2020], we changed the success criteria for ADD(S) to 0.05, 0.10(original) and 0.15 of the target objects radius. The smaller values for the success criteria will penalize the performance of a given method as it will count

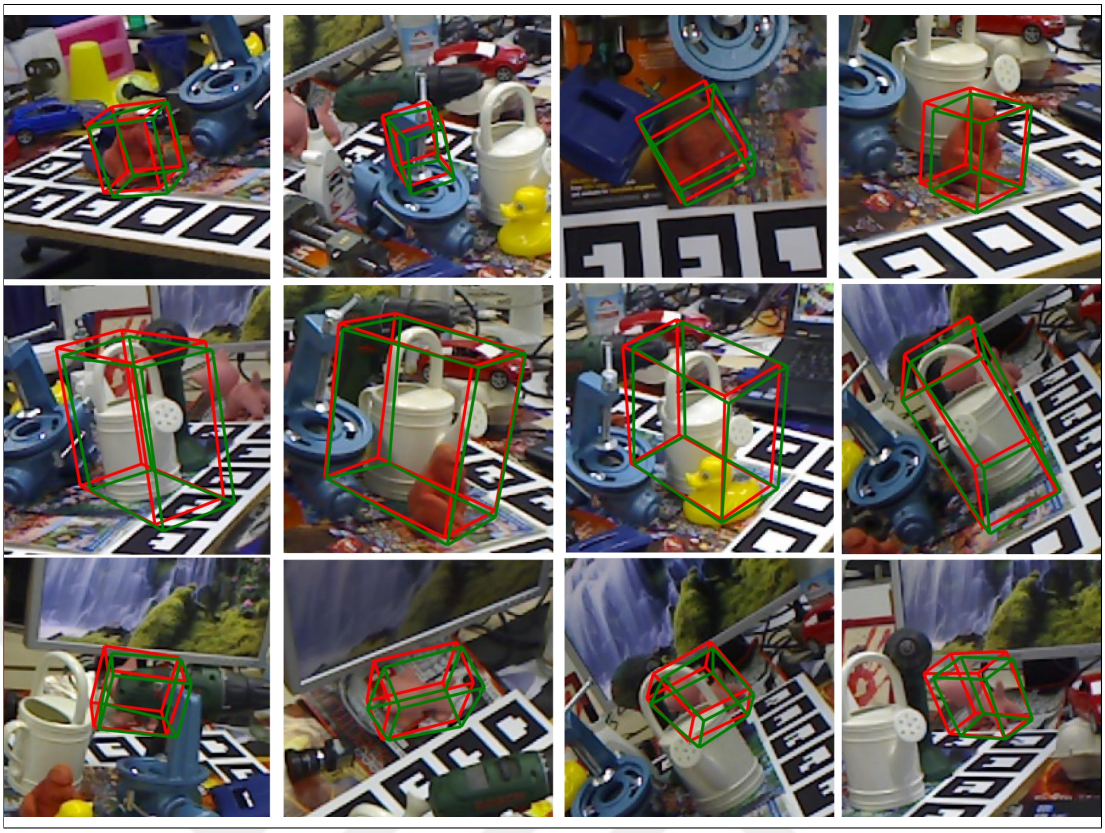

Figure 4.6: Qualitative results on estimated poses on Occlusion LINEMOD dataset.

smaller errors as failures.

We observe that the aspect prediction adds a significant improvement when smaller success criteria is applied. This is in part expected since aspect classification constraints the key-point selection process making it more precise. In other words, without the aspect predictor, the network is not constrained enough to stay within the bounds of the aspect. With aspect predictor, initial estimate of the pose falls within a certain range, and key-points selection is partially constrained to be selected in this range. Additionally, 3D pose estimation is done with respect to a coordinate frame and choosing a coordinate frame for each aspect lets the PnP solver to search for and find small displacements (or object motions). PnP algorithms tend to do better when the target motion is smaller.

Pose refinement stage always improves the results as observed by [Li et al., 2020]. This is partially due to the fact that the object silhouette is used. Silhouette is a strong cue when the object has very little texture. Aspect classifier makes use of the silhouette

as well as the apparent texture of the object. However, the aspect classifier does not return a full pose but a close one. Silhouette matching would improve this discrete pose as seen in these results. We further observe that the relative improvement that aspect classifier brings is much larger for some difficult objects such as ape and cat when the success criteria is reduced.

In Fig. 4.4, we present a detailed analysis of the contribution of aspect proposal and post-refinement stages using the 2D projection metric. Our analysis suggests that aspect proposal and refinement stages have a positive impact on estimation on every object class. Another observation is that the impact of these additional stages are more prominent when the bare-bone method fails. As the threshold increases perceived accuracy on different models converges as expected.

4.8.1. Examination of number of view-points and choice of polyhedron

We expand our experiments by applying a different number of viewpoints. We use the LINEMOD dataset and previous setup as a baseline. The number of view points is selected from a set of (16, 32, 64, 96). In table Table 4.6 we show the contribution of different number of view points. We observe that low number of viewpoints have significant negative effect on object that have similar dimensions. On the other hand, number of viewpoints have diminished returns on this polyhedron.

The second part of the experiment focuses on different polyhedra on multiple objects. We used Blender's Convex Hull algorithm for generating encompassing a particular set of objects from the LINEMOD dataset. (ape, bench, driller, and duck). Similar to the original setup we build 64 viewpoints. However, in this case, we had to change the visibility algorithm in order to compensate for the irregular convex hull. We assume a face is visible if a casted ray from the camera center hits at least two vertices of a particular face. Then we follow the same algorithm to construct the psuedo-aspect graph of the target objects and use the same training strategy. Our reason behind this selection is that ape and duck object clearly benefits from our pipeline.Furthermore, our method clearly performs worse on bench and driller objects consistently compared to

Table 4.7: Comparison of regular polyhedron and convex hull as delegate representations for constructing aspect graph.

Objects	ape	bench	driller	duck
Ours(Icosahedron)	82.2	94.0	95.4	94.0
Ours(Convex Hull)	76.4	92.1	93.7	89.9

other studies. We show the results of the second study in Table 4.7. Results suggest that, there are some issues with convex hull. We believe that our crude visibility algorithm is not a appropriate choice for convex hull based aspect-graph generation.

4.8.2. Run-time analysis

Given an image of size 640×480, our method estimate the segmentation map and dense keypoint vectors in 8.1ms. Key-point location takes 29 ms on average. And PnP takes around 2.5ms. Total inference time for a single pose is around 39 ms for a single object on a single thread i5 machine with RTX 2060 Super GPU using python. A C++ based multi-threaded(4 threads) variant of the same pipeline cuts key-point inference to 29ms(average) and PnP to 2ms. resulting 29ms average per scene.

4.9. Summary

In this chapter, we show that adding a classifier block estimating the predefined aspects of the objects improves the multi-stage process. This is due to the fact that the additional classifier acts as a constraint simplifying the required neural network and at the same time yielding better key-point selection. We reduce the search space for the key-point selection and exclude false-positives by mapping the appearance of an object to an aspect. The simplified neural network allows faster inference and a smaller footprint.

5. IMPROVING 6-DoF POSE ESTIMATON PERFORMANCE USING CURRICULUM LEARNING

This chapter explores the idea of improving training process itself. We start training a pre-trained lightweight neural network with relatively easy poses with small or no occlusion. As the training progress, we introduce noise and smart occlusions driven by priori key-point information and backpropagated attention. Our observations suggest that the proposed hierarchical learning strategy helps the neural network to generalize better compared to a traditional training strategy. Additionally, the performance of the generated model is on par with the state-of-the-art models albeit having a smaller computational and memory footprint.

5.1. Introduction

Object pose estimation recovers the orientation and translation of an object within a given view with respect to a reference coordinate frame. It is an essential part of a successful application in augmented reality, robotics and autonomous vehicles. For instance, in an augmented reality setting, a user's interactions with an object are heavily depended on its pose. If the object pose is incorrect, any augmentation will look unnatural. In robotics, a robot must estimate the pose of the targets so that it can perform tasks such as grasping and relocating. Pose inference from a single RGB image can be very challenging. Many of the challenges come from the scene itself, such as background clutter, varying lighting and imaging conditions. Other challenges are due to the object itself, like self-occlusion, lack of texture or texture changes due to wear and tear.

Traditionally, pose estimation problems are solved by corresponding low-level features between the image and object. These features are usually hand-crafted based on edges, corners or small contours [Lepetit and Fua, 2005]. They do not include global information (such as object topology) or the semantics of the scene. Furthermore, these

techniques do not cope with textureless objects well. Since underlying feature selection algorithms [Pong and Cham, 2006] usually require rich textures on the object to build robust features. Finally, these solutions perform poorly under challenging set-ups like occlusion or scene clutter. Recently deep-learning researchers have proposed a broad range of algorithms [Gupta et al., 2019; Peng et al., 2019; Li et al., 2018] to tackle the pose estimation problem. These methods seem to capture object representations even when there is a lack of texture. They can learn object semantics (related to aspects or viewpoints of the object) through a significant amount of training. Some of these studies, either directly estimate the object pose using a classification or regression scheme. While, the rest use deep pre-trained convolutional networks (CNN) as high-quality semantic feature selectors and apply a robust (e.g., random sample consensus algorithm - RANSAC) pose estimation algorithm (e.g., 3D to 2D perspective n-point PnP).

With the help of deep learning, especially its ability to handle a tremendous amount of training data, researchers overcome most of the obstacles seen in traditional pose estimation techniques. However, scene clutter and occlusion remain some of the degrading factors of the pose estimator's performance. Recent research suggests alternative approaches like using dense representations, object masks to constraint and guide the network to focus on both global and local features to overcome the previously mentioned problem. Yet this solution comes with multi-stage estimations with large network sizes and additional computational burden.

In this manuscript, we propose an adaptive training regime that requires no computational overhead during inference time. We employ curriculum learning by progressively applying random masks driven by keypoint locations and attention on the object surface during training. We start by using simple poses(simple scenes with no occlusion) as the training progresses, we selectively introduce masks with random color and shapes with growing numbers and areas. In the final stages of training, we estimate the auxiliary information sources by backpropagating the gradient on the object. Using this information, we occlude some of the attended features. While this strategy adds

an lengthy overhaul to the training process, we benefit from a fast and accurate pose estimation during inference.

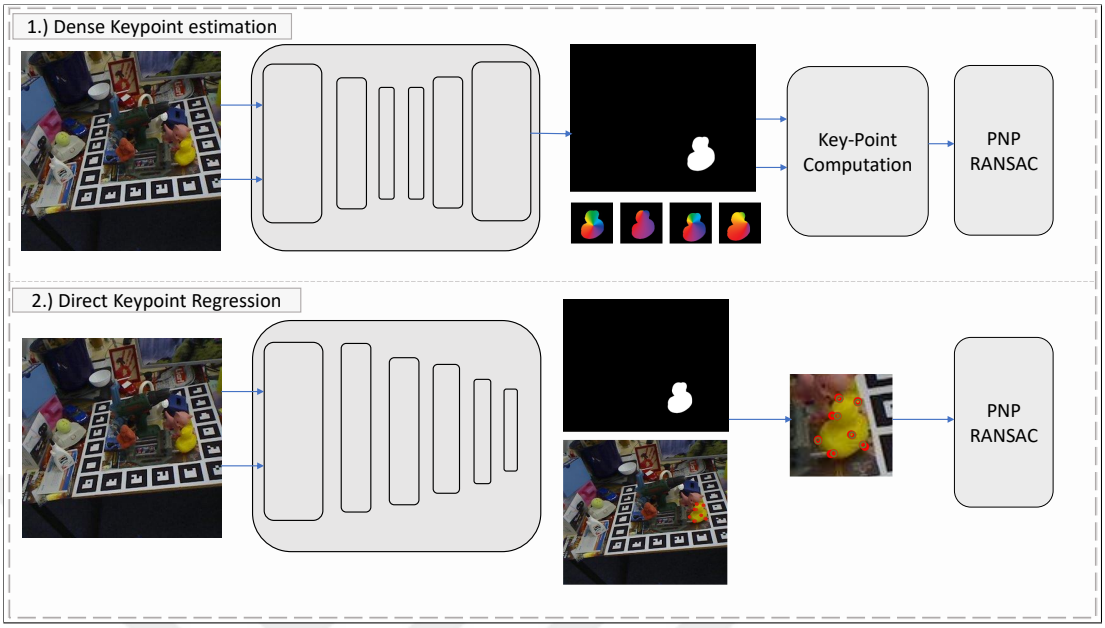

Figure 5.1: Curriculum Learning pipelines.

We summarize our contributions as:

- We introduce a sound curriculum strategy for pose estimation. Our curriculum incorporates advantage of different sample sources. Additionally we demonstrate that, constructing a curriculum strategy yields better results than plain usage of data samples.
- The proposed pipeline is significantly faster and comparable in accuracy to the state-of-the-art pose estimation algorithms. We demonstrate this is the result of backbone architecture and reduction in the size of key-point space during estimation.
- We utilize pose-perturbations which we call pose groups and include artificially generated smart occlusions similar to an adversarial training methodology.

5.2. Related Work

5.2.1. Pose Estimation

Contemporary pose estimation approaches heavily rely on deep learning(DL) like other fields of computer vision. Modern neural network's ability to utilize lots of training data made researchers leave traditional local feature construction-based detection schemes. In one of the first implementations [Xiang et al., 2018] employ a CNN architecture to estimate 6-DoF pose by regressing to a quaternion representation. However, their method heavily suffers when the object is occluded. Moreover, [Oberweger et al., 2018] show that occluders havea corrupting effect on CNN activations far beyond thereceptive field. In response to those failures, many researchers adopt a multi-stage pipeline similar to the traditional methods with one key difference. Instead of using hand-crafted features, they employ neural networks spesifically, CNN's to do the heavy-lifting of key-point estimation, followed by a 2D-3D key-point matching. [Rad and Lepetit, 2017] uses segmentation and sparse correspondences for detecting the 3D bounding box. [Tekin et al., 2018] use YOLO [Redmon et al., 2016] architecture to estimate object key-points on a multi-resolution feature map. In a dense key-point representation approach, an alternative to sparse key-point matching, a few selected key-points are estimated in a dense-approach in which every object pixel has a contribution to the pose estimation. Similar to a Hough voting scheme, all pixels cast a vote to a set of hypotheses. The hypothesis with the most votes are selected to be the key-point proposals. [Kehl et al., 2017] use CNN to sample RGB-D image patches and extract features for the voting. In another dense approach, [Peng et al., 2019] uses the Farthest Point Sampling algorithm (FPS) to select key-points on the target object. then encodes a direction vector from the object pixels to the selected key-points. The network is trained by minimizing the distance between target vectors and the predicted vectors. Finally, they apply an uncertainty driven PnP algorithm to determine the final object pose.

5.2.2. Curriculum Learning

Introduced by [Elman, 1993], named and expanded by [Bengio et al., 2009], *Curriculum Learning* imitates nature of human learning process. In this strategy, a given task samples are re-organized into a meaningful order that gradually increase the complexity of the task. Researches have demonstrated the effectiveness of this ordering yields better results than the ubiquitous stochastic mini-batch in neural machine translation [Xu et al., 2020], natural language understanding [Graves et al., 2017], image recognition [Hacohen and Weinshall, 2019]. The main challenge of curriculum learning is to differentiating and organizing easy and hard samples. There is a plethora of published research regarding this *sample mining* which are very well covered in [Soviany et al., 2021]. Curious readers may refer to the survey to gain depth on the subject.

5.2.3. Interpretability and saliency in deep learning

Despite the tremendous success of DL, It partially remains as a black-box decision maker. Research in saliency and interpretability aims to explain decisions of a neural network by analyzing using but not limited to signal pathways [Shrikumar et al., 2017], response(activation) strength [Zhou et al., 2016] and gradient analysis [Selvaraju et al., 2017]. In our scenario we employ a basic saliency and activation mapping technique to introduce guided perturbations and occlusions into our samples to generate hard samples. In the next section, we will cover the details of used techniques for constructing organized curricula for pose estimation.

5.3. Empirical Studies

5.3.1. Problem definition

We are looking for accurate, light-weight pose estimation machinery which can run near real-time on mobile devices. Recent approaches incorporate multi-stage pose estimation and inference refinement, techniques to the pipeline. Additionally, they

introduce architectural changes and explicit structures to benefit from the additional information like object geometry. These changes undeniably improve the overall performance of the algorithms. However, these improvements cost a considerable amount of computational power and degradation in estimation duration. Considering our constraints, we develop a two-stage pose estimation pipeline. The first stage is a pre-trained neural network MobileNet, designed to run on mobile devices. We tailor the network to regress key-point locations on a target image. Followed by a single-stage PnP algorithm [Lepetit et al., 2009] to estimate the final object pose. We employ a curriculum learning strategy to alleviate the diminishing accuracy. In the next section, we cover the details of the curriculum construction.

5.3.2. Curriculum Construction

In this section, we present our tools and strategies for establishing a sound curriculum. In most computer vision tasks, constructing a meaningful curriculum is a challenge. It usually requires a pre-training, or a parallel mining operation to differentiate easy samples from hard samples. Moreover, there are also challenges when composing batch scheduling with consistent in-batch statistics.These operations are usually computationally expensive and require analysis of the effects of individual samples. On the other hand, domain knowledge may help to understand or even manufacture samples with controllable levels of complexity, such as in our case. In pose estimation problems, occlusion, scene clutter, and object illumination are some of the challenges that degrade the performance of algorithms. Using this information, we build our curriculum using two different approaches. In the first approach, we directly synthesize images using 3D rendering software. We place the target objects into scenes with no background and render them on the backgrounds from the MS-COCO dataset, and finally in the scenes with heavy clutter with a natural-looking environment with clutter and heavy occlusion.

Furthermore, we randomly perturb the object pose by a few degrees $(< 3^\circ, < 3cm)$ two more times to construct a pose group. We will refer back to this pose group while

Figure 5.2: An example of viusal pose group.

talking about hard sample mining. We place the generated images into sets depending on their perceived hardness (depending on the object visibility and background complexity). These images are sampled throughout the training and validation process depending on the epoch number. In the second strategy, we use an online method to develop complex samples. At first, we introduce crops with random shapes, colors around the key points. The size of the crops depends on the projected size of the target on the object. In addition, we enlarge the cropped area as the training progresses. Another source of dynamic samples comes from the saliency and attention of the neural network. In the latter stages of training, we calculate per-sample attended areas of the target image and apply a similar occlusion around the target. Final dynamic samples come from the pose group discussed before. We consider a posed group a correct (consistent) estimation, if the pose predictions of all samples lie within a certain threshold. During the initial stages of the training, we randomly select one of the samples in the group for batch construction. As the training advances, we estimate the pose of all samples in a group. If the results are inconsistent, one sample is selected from the group and included in the batch.

5.3.3. Experimental setup

We train our network to estimate normalized image coordinates of the key-points. For each target object, we randomly select 18 key-points from the vertices of the 3D model and 1 key-point representing the center of the object. A total number of 19 point is used in training. We use a combined L_1 and L_2 loss as objective function.

We use a pre-trained MobileNet [Sandler et al., 2018] implementation in TensorFlow2 [Matthews et al., 2017] as neural backbone. Our model is optimized using Adam [Kingma and Ba, 2015] for 200 epochs. We use a dynamic batch size approach in our approach which we will cover the details in the next section. We augment the training data slightly by randomly perturbing hue, saturation and exposure.

5.4. Datasets and Performance Evaluation

We train our network on well-known LINEMOD [Hinterstoisser et al., 2013].This dataset is highly challenging due to background clutter, variations in illumination and texture-less objects. This dataset includes 1200 images of 13 objects with annotated poses with clear train-test split. In our experiments we do not use any of the training samples. For each of the 13 objects, we generate 6000 image pose-groups(total of 18000) whose object poses are sampled uniformly based on the distribution of the corresponding dataset. A third of the images are rendered on a random uniform background, Another one third is synthesised on the random images from PASCAL-VOC dataset [Everingham et al., 2015] using the approach described in [Dwibedi et al., 2017]. The final third is rendered on a pre-modeled 3D-scene. We use the scene as a natural looking clutter and occlusion environment. We have also tested our hypothesis on Occulison LINEMOD [Brachmann et al., 2014] dataset.Occulison LINEMOD is a variation of the original LINEMOD dataset. 8 objects in the LINEMOD dataset are additionally annotated. In total 1215 frames with heavy occlusion are selected from the original dataset.

We follow the literature to evaluate the performance of 6-DoF algorithms. 2D Projection. The 2D Projection metric measures how close the 2D projected vertices are to the ground-truth, in pixel domain. Following the common practice in the recent papers, we consider the pose as correct if the mean 2D projection error is below 5 *pixels*. ADD Metric Average Distance (ADD) computes the mean of the pairwise distances

between the ground truth pose and the estimated pose:

$$ADD = \frac{1}{N} \sum_{x \in M} \|(\hat{R}x + \hat{t}) - (Rx + t)\|_2 \tag{5.1}$$

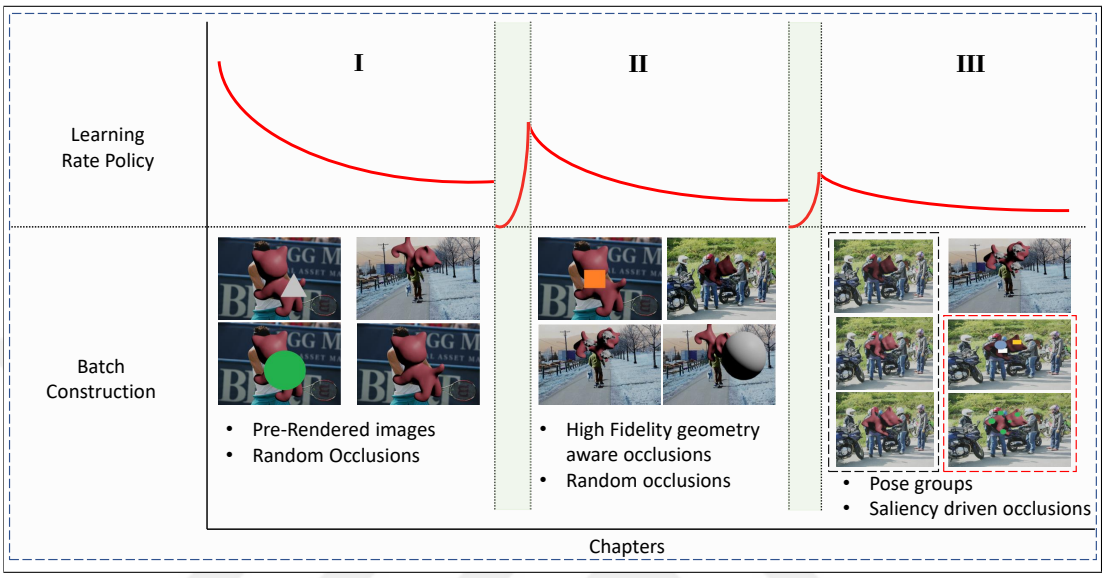

Figure 5.3: Visualization of studied curriculum strategy.

where $\hat{R}$,$\hat{t}$ are predicted rotation and translation R,t are ground truth labels. M describes the set of model points, N defines the size of the set. The 6-DoF pose is considered correct if the ADD score is smaller than a threshold. This threshold is not a constant value and depends on the size of the target object. Following the previous works, we set this threshold to be 10% of the target objects diameter.

5.4.1. Training and Batch Construction

We divide the overall training into three phases, which we will refer to these as ”Chapters” from now on. The first and last chapter takes 60 epochs the second chapter is 80 epochs long. Each chapter has different levels of ”hardness” depending on the information source, occlusion range, and type. Additionally, each chapter has different learning rates. We decay the learning rate exponentially by a rate of 0.98 in the middle of each chapter to minimize the risk of forgetting. Finally, we increase the batch size at the beginning of each chapter to include the previous chapter’s examples to increase

Table 5.1: Comparison of 2D projection accuracy on LINEMOD dataset.

Object	BB8	Tekin	PvNET	CullNet	Ours
ape	95.3	92.10	99.23	96.6	98.2
bench	80.0	95.06	99.81	90.1	93.7
cam	80.9	93.24	99.21	86.0	95.7
can	84.1	97.44	99.90	91.2	98.1
cat	97.0	97.41	99.30	98.8	94.2
driller	74.1	79.41	96.92	80.9	96.3
duck	81.2	94.65	98.02	92.2	95.7
eggbox	87.9	90.33	99.34	91.0	98.0
glue	89.0	96.53	98.45	92.3	97.4
hpuncher	90.5	92.86	100.0	95.3	99.0
iron	78.9	82.94	99.18	84.8	96.9
lamp	74.4	76.87	98.27	75.8	94.2
phone	77.6	86.07	99.42	85.3	98.0
average	83.9	90.37	99.00	89.3	89.0

resilience.

In the first chapter, we sample the batch elements from the images with the uniform background and the images placed on top of the PASCAL-VOC dataset. We select the learning rate of 0.001 and batch size 16. We procedurally occlude key-points described in section 3.2 until the chapter ends. In the second chapter, we adjust the learning rate to 0.006 and increase the batch size to 32. We compose half of the batch from the first chapter. We select the rest of the batch from the high-fidelity renderings of natural scenes created using the Junk Shop scene created by Alex Treviño and Anaïs Maamar in Blender. In this chapter, we apply occlusions and clutter by using the 3D models given in the dataset and we use a procedural placement of 3D objects between the objects and the camera. In the final chapter of training, we set the learning rate to 0.001. We keep the batch size the same and use the same image sources. The main difference, however, is the construction of each batch. Besides the online masking, we actively search and mine for hard samples using attention and pose groups. One third of the batch elements come from random masking of the key-points. Another one third selected similar to an adversarial training strategy, however in this training scheme we do not pursue the adversarial examples by gradient projection, rather calculate the saliency maps and apply random occlusions around the informative regions. For

Table 5.2: Comparison of ADD(S) accuracies on LINEMOD dataset.

Object	BB8	SSD-6D	Tekin	PvNET	CullNet	Ours
ape	27.9	2.6	21.6	43.62	55.1	52.4
bench	97.5	15.1	81.8	99.90	89.0	91.0
cam	40.1	6.1	36.6	86.86	66.2	88.4
can	48.1	27.3	68.8	95.47	89.2	92.8
cat	45.2	9.3	41.8	79.34	62.6	76.5
driller	58.6	12.0	63.5	96.43	88.6	88.6
duck	32.8	1.3	27.2	52.58	44.3	85.4
eggbox	40.0	2.8	69.6	99.15	97.1	96.4
glue	27.0	3.4	80.0	95.66	94.6	94.8
hole	42.4	3.1	42.6	81.92	68.9	71.3
iron	67.0	14.6	74.9	98.88	90.9	96.6
lamp	39.9	11.4	71.1	99.3	76.5	86.1
phone	35.2	9.7	47.73	92.41	54.0	94.0
average	43.6	9.1	55.9	86.27	78.3	85.7

each image, we compute the saliency maps for a random subset of key-points by back-propagating the gradient on to target image. This is followed by applying a mask in the center of the heat map. The size of the mask is calculated using the 2D bounding box of the target object. The final source of this chapter is again the pose-groups. we predict the pose for each object in the group. If the average error is greater than 4 cm in translation or 4° in rotation. We include a random element from this set in the batch. We train the a model per target object. A training session lasts around 24-26 hours. We use Blender for rendering the training images and computing projection matrices and labels. We performed our experiments on an RTX 2060 Super GPU with 32 GB of RAM and a RTX 2080 with 16GB of RAM. At the beginning of of each chapter we apply a warm-up period for 1 epoch and gradually increase the learning rate from zero to the determined value. Then we apply a learning rate decay until it is halved at the end of each chapter.

5.5. Results

We show the accuracy on 2D projection metric on LINEMOD dataset in Table 5.1. Our method performs similar to the state of the art despite having a much smaller

Table 5.3: Comparison of ADD(S) metric accuracies on Occlusion datasets

Object	Tekin	PoseCNN	PvNET	CullNet	Brynte	DeepIM	Ours
ape	2.48	9.6	15.81	21.97	40.85	59.18	32.1
can	17.48	45.2	63.30	24.52	82.44	63.52	64.2
cat	0.67	0.93	16.68	9.77	35.64	26.24	28.6
driller	7.66	41.4	25.24	23.62	71.33	55.58	54.3
duck	1.14	19.6	65.65	26.11	49.08	52.41	65.0
eggbox	-	22	50.17	20.43	57.28	62.95	53.0
glue	10.08	38.5	49.62	28.02	62.90	71.66	48.6
hole	5.45	22.1	39.67	41.4	43.14	52.48	44.3
average	6.42	24.9	40.77	24.48	55.33	55.50	48.8

foot-print and less supervision(sparse key-points vs dense key-point representations in the PV-NET and lack of natural scenes). In addition, Table 5.2 demonstrates our results based on ADD(S) metric. Proposed pipeline is again comparable to the state-of-the-art using synthetic data supervision. Due to diminished scene clutter and less Occlusion competitive models perform quite well on the LINEMOD dataset. The extended Occlusion LINEMOD dataset is much challenging as the name suggests. We present our our findings in the Table 5.3. Additional experiments on this dataset validates effectiveness of our method. As we will show in the ablation studies, inclusion of a curriculum strategy benefits the algorithm similar to a post refinement process.

5.5.1. Ablation studies

In this section we present a detailed ablation study. We compared the contribution of each design choice to the overall results. Furthermore, we study the general behaviour of our training strategy on different algorithms in the contemporary research.

5.5.2. Dense key-point estimation vs Direct regression

In figure 5.1, we outline two different strategy for key-point location estimation. the first pipeline, key points are determined by estimating direction vectors for each object pixel. These direction vectors point towards the pre-selected key-points. Then these direction vectors are transformed to key points by a voting heavy-weight voting mechanism. In the second strategy we directly regress the key-point locations. Both

Table 5.4: Comparision of each training chapter to the overall accuracy (ADD(S)) of each chapter on OCCLUSION LINEMOD dataset.

Object	Chp. 1	Chp 2.	Chp. 3	Mixed
ape	12.1	16.1	15.2	27.6
can	23.4	28.4	24.9	43.6
cat	21.8	29.4	26.3	32.3
driller	21.4	26.4	22.2	36.4
duck	16.4	33.8	33.7	47.3
eggbox	17.1	25.3	21.3	34.7
glue	16.0	28.5	28.8	35.4
hpuncher	14.4	31.2	28.4	38.2
average	17.8	27.3	25.1	36.9

strategies use a robust method to estimate the final object pose. Table 5.5 shows the the accuracy comparison on Occlusion LINEMOD dataset. Dense key-point estimation is an extra-ordinary benefactor due to the explicit representation of key-point and object geometry as a whole. However, the main drawback is that in order to calculate the key points a post calculation process is required and additional computational power is consumed while estimating the general key-point directions. Our approach closes the performance gap between dense and direct estimation methods while execution time is almost one-third.

5.5.3. Contribution of learning chapters

In this section, we perform two ablation studies. The first one is straightforward. We record the best accuracy metric for each chapter. In the second study, We construct four different experimental setups(3 for each chapter and one for directly mixing the data). In these setups, each chapter has the same amount of training images. These images cover the same pose spaces from the original setup. Each chapter is expanded in a way such that number of images and the covered posed space is similar to the original setup. Furthermore, scene clutter (occlusions and perturbations) is introduced using the same algorithms. Table 5.4 shows the result the results for our ablation studies. In column 2 we clearly see the advantage of incorporating high-fidelity images, while accuracy gain is considerable, it is not near the state of the art. The Mixed training

Table 5.5: Comparison of ADD(S) metric accuracy of Direct Key-point regression vs, Dense key-point vectors accuracy on Occlusion LINEMOD dataset.

Object	Ours (Dense)	Ours (Regression)
ape	32.1	26.3
can	64.2	59.4
cat	28.6	28.0
driller	54.3	52.1
duck	65.0	58.1
eggbox	53.0	41.4
glue	48.6	42.4
hpuncher	44.3	45.7
average	53.4	51.9
run time	37ms	14 ms

performs significantly better than rest of the strategies. Yet, performance of the proposed method is far worse than systematic inclusion of training samples as shown in Table 5.3. We believe that there are several reasons for this observations. First of all, each training chapter provide a consistent sampling serve a good initialization point for the following one. Thus allowing the network to search for a better minima in a constraint space. Secondly, it is known that, statistically inconsistent training samples can impede the learning process causing longer training times and over-fitting even in relatively simple models. Since underlying statistics of each chapter is fundamentally different from each other, this can cause numerical instabilities across batches during training. By grouping(in a perceived) similar samples, we provide a intrinsic regulative behavior to the training process.

5.5.4. Curriculum strategy on other studies

We use the same training strategy with same dataset on PvNET[Peng et al., 2019] We choose this study due to several reasons. First and foremost, it that without any refinement process it produces state of the art results using this dataset. Secondly, implementation of the algorithm is straight-forward and easy to understand. Finally, the amount of synthetic data used in the training is similar to our training data. Table 5.6 demonstrates the final results from the study. Our experiments show the significant

Table 5.6: Performance of proposed training strategy on PvNET model.

Object	PvNET	PvNET with Curr.
ape	15.81	30.3
can	63.30	77.3
cat	16.68	46.7
driller	25.24	58.3
duck	65.65	71.4
eggbox	60.17	57.0
glue	49.62	56.7
hpuncher	39.67	48.2
average	40.77	55.7

improvement on every object class.

5.5.5. Contribution of pose groups

In initial experiments, we observe that most of the time, neural models predict the key-point locations relatively well, even under occluded scenes. Furthermore, additional training data does have a significant diminishing returns. When we analyze the predictions, we observe that, when network fails to predict correct key-points, the error becomes very large. In other words, our method degrade catastrophically when it make prediction errors. While combating this we notice a large deviations in the predictions when a small amount perturbation is applied on to the original poses. In the final chapters of training, we incorporate pose groups in to the training regime. Pose groups help to recover almost half of the catastrophic predictions. However, we observe an average drop (1.3 point in ADD(s) metric) in LINEMOD dataset in previously consistent predictions. We believe that pose groups relax the constraints of object appearance similar to using a noisy data or labels. It expands the underlying prediction space. This removes point-to-point relation between object appearance and pose and enables and area-to-area mapping, from a geometrical point of view. On the other hand, this results in a larger intra-pose-space deviation since the point constraint is removed.

5.6. Summary

In this chapter, we introduced a curriculum based training strategy for 6-DoF pose estimation. During training distribution of training data is shifted from trivial examples to increasingly challenging ones. In order to construct a sound strategy, we have employed data augmentation, attention, adversarial examples and a relaxation factor called pose groups. Pose-groups treats a number of close poses as a single pose and assigns the same label to each of the sample.

6. CONCLUSION

This study presents novel 6-DoF pose estimation methods to remove some of the obstacles intrinsically connected to the deep learning such as, heavy computational requirements, dependency on expensive data annotation procedures and, guiding and enforcing priori knowledge into the training procedure.

In the second chapter, We have presented a method for direct estimation of the camera pose from a given image of known objects. Starting with a CAD model of the object for which the pose to be estimated, we first render synthetic images of the object under varying pose, occlusion and illumination conditions. Augmenting these synthetic images with natural background yields a training data. This data is then used to train a carefully designed neural network. This model estimates the camera pose in our data better than any other direct method that we know. The same model performs comparable to the other methods on benchmark datasets such as Cambridge and 7Scenes. The proposed model is fine-tuned for this, purpose with a specific architecture which is much smaller than many reported in the literature. We also introduced the random projective loss function that helps the performance especially in the orientation estimation by 1^o. The exponential regularizer we have used also helps in the performance of both our model as PoseNet Kendall et al. [2015]. Combined, these three improvements in our model leads to a fast and accurate direct pose estimation method.

In the third chapter, we proposed a novel pipeline for real-time 6DoF pose estimation. Our pipeline uses aspects based on object appearance and dense key-point predictions for pose calculation. We use a recent off-the-shelf PnP solver for final pose estimation. We show that including an appearance classifier into the pose estimation process outperforms most of the approaches in the literature. We also show that our pose estimations can further be improved by a post-refinement stage.

In the final chapter, we proposed a novel training strategy with detailed training steps called chapters for real-time 6-DoF pose estimation. Our method directly predicts pre-selected key-points and estimates the object pose using a robust method. In our

studies, we demonstrate that using a carefully constructed curriculum helps the pose estimation process significantly. We also detail the rational of the curriculum strategy by dissecting each chapter. The performance of the method is also demonstrated using other studies. Our studies suggest that using a curriculum strategy enable light-weight neural networks perform comparable to the state-of-the art methods enabling accurate and fast inference in computationally limited devices like cellphones.

6.1. Future Work

Firstly, we are currently investigating a patch based pose estimation solution. In this scenario, a few local patches of the object will be queried to a implicit representation of the target object. This representation and pose estimation pipeline will be jointly trained in a cycle-consistent manner. we will also focus on refining the aspect-generation process, with graph and machine learning-based algorithms. As our early studies has shown, the granularity of the aspect seems to make a difference in performance. We plan to investigate alternative aspect generation techniques and their impact on the propose object pose estimation methods.

REFERENCES

Abadi M., Barham P., Chen J., Chen Z., Davis A., Dean J., Devin M., Ghemawat S., et al. (2016) ,"Tensorflow: A system for large-scale machine learning" ,In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah. USA, 2-4 November.

Akgul O., Penekli H. I., Genc Y. (2016) ,"Applying deep learning in augmented reality tracking" ,In International Conference on Signal-Image Technology and Internet-Based Systems (SITIS), 47–54, Naples. Italy, 28-30 November.

Bengio Y., Louradour J., Collobert R., Weston J. (2009) ,"Curriculum learning" ,In ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, 41–48, Montreal, Quebec, Canada. 11-16 July.

Brachmann E., Krull A., Michel F., Gumhold S., Shotton J., Rother C. (2014) , "Learning 6d object pose estimation using 3d object coordinates" ,In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 8690 LNCS, 536–551. Springer Verlag.

Choi J., Chun D., Kim H., Lee H. J. . (2019) ,"Gaussian yolov3: An accurate and and fast object detector using localization uncertainty for autonomous driving" ,In Proceedings of the IEEE International Conference on Computer Vision, 502–511, Seoul, Korea. 27-30 October.

Dede M. A., Aptoula E., Genc Y. (2019a) ,"Can additional spectral bands be estimated from aerial color images?" ,Turkish Journal of Electrical Engineering and Computer Sciences, 27(3):1696–1703.

Dede M. A., Aptoula E., Genc Y. (2019b) ,"Deep network ensembles for aerial scene classification" ,IEEE Geoscience and Remote Sensing Letters, 16(5):732–735.

Doumanoglou A., Kouskouridas R., Malassiotis, S. K. T. (2016) ,"Recovering 6d object pose and predicting next-best- view in the crowd" ,In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3583–3592, Las Vegas, Nevada. 26-30 June.

Dwibedi D., Misra I., Hebert M. (2017) ,"Cut, paste and learn: Surprisingly easy synthesis for instance detection" ,In Proceedings of the IEEE International Conference on Computer Vision, 1301–1310, Venice, Italy. 8-13 October.

E. Hinterstoisser, A. Krull F. M., Gumhold S., Shotton J., et al. (2014) ,“Eccv” ,In Learning 6D object pose estimation using 3D object coordinates, 536–551, Munich, Germany. Springer, Cham, 8 - 14 September.

Elman J. L. (1993) ,“Learning and development in neural networks: the importance of starting small” ,Cognition, 48(1):71–99.

Everingham M., Eslami S. M. A., Van Gool L., Williams C. K. I., Winn J., Zisserman A. (2015) ,“The pascal visual object classes challenge: A retrospective” ,International Journal of Computer Vision, 111(1):98–136.

Everingham M., Van Gool L., Williams C. K. I., et al. (2010) ,“The pascal visual object classes (voc) challenge” ,International Journal of Computer Vision, 88:303–338.

Faugeras O., Mundy J., Ahuja N., Dyer C., Pentland A. e. A. (1992) ,“Why aspect graphs are not (yet) practical for computer vision?” ,CVIGP, 55:212–218.

Fischler M. A. Bolles R. C. (1981) ,“Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography” , Commun. ACM, 24(6):381–395.

Forsyth D. A. Ponce J. (2002) ,“Computer Vision: A Modern Approach” ,Prentice Hall Professional Technical Reference.

Gao X.-S., Hou X.-R., Tang J., Cheng H.-F. (2003) ,“Complete solution classification for the perspective-three-point problem” ,IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(8):930–943.

Girshick R. (2015) ,“Fast r-cnn” ,In 2015 IEEE International Conference on Computer Vision (ICCV), 1440–1448, Santiago, Chile. 8-13 December.

Graves A., Bellemare M. G., Menick J., Munos R., Kavukcuoglu K. (2017) , “Automated curriculum learning for neural networks” ,In Proceedings of the 34th International Conference on Machine Learning, ICML’17, 1311–1320, Sydney, NSW, Australia. JMLR.org, 20-24 June.

Gupta K., Petersson L., Hartley R. (2019) ,“Cullnet: Calibrated and pose aware confidence scores for object pose estimation” ,In Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019, 2758–2766, Seoul, South Korea. 27 Oct – 2 Nov.

Hacohen G. Weinshall D. (2019) ,“On the power of curriculum learning in training

deep networks” ,ArXiv, abs/1904.03626.

Haralick R., Lee D., Ottenburg K., Nolle M. (1991) ,“Analysis and solutions of the three point perspective pose estimation problem” ,In Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 592–598, Maui,Hawaii, USA. 3-6 June.

Hartley R. Zisserman A. (2003) ,“Multiple View Geometry in Computer Vision” , Cambridge University Press, New York, NY, USA, 2 edition.

He K., Zhang X., Ren S., Sun J. (2016) ,“Deep residual learning for image recognition” ,In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778, Las Vegas, Nevada. 26-30 June.

Hess R. (2010) ,“An open-source sift library” ,In MM’10 - Proceedings of the ACM Multimedia 2010 International Conference, Firenze, Italy. 25-29 October.

Hinterstoisser S., Lepetit V., Ilic S., Holzer S., Bradski G., Konolige K., Navab N. (2013) ,“Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes” ,In Computer Vision – ACCV, 548–562. Springer Berlin Heidelberg.

Hodan T., Michel F., Brachmann E., Kehl W., GlentBuch A., Kraft D., Drost B., Vidal J., Ihrke S., Zabulis X., Sahin C., Manhardt F., Tombari F., et al. (2018) ,“Bop: Benchmark for 6d object pose estimation” ,In Proceedings of the European Conference on Computer Vision (ECCV), 19–34, Munich, Germany. 8 - 14 September.

Hodaň T., Haluza P., Štěpán Obdrzalek, Matas J., Lourakis M., Zabulis X. (2017) , “T-less: An rgb-d dataset for 6d pose estimation of texture-less objects” ,In Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017, Santa Rosa, CA,USA. 24-31 March.

Huang G., Liu Z., Weinberger K. Q. (2016) ,“Densely connected convolutional networks” ,In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2261–2269, Las Vegas, Nevada. 26-30 June.

Kehl W., Manhardt F., Tombari F., Ilic S., Navab N. (2017) ,“Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again” ,In Proceedings of the IEEE International Conference on Computer Vision, 1530–1538, Venice, Italy. Institute of Electrical and Electronics Engineers Inc., 22-29 October.

Kehl W., Milletari F., Tombari F., Ilic S., Navab N. (2016) ,“Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation” ,In ECCV, Zurich, Switzerland. 23-28 August.

Kendall A. Cipolla R. (2017) ,“Geometric loss functions for camera pose regression with deep learning” ,In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, USA. 22-25 July.

Kendall A. Gal Y. (2017) ,“What uncertainties do we need in bayesian deep learning for computer vision?” ,In Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R., editors, Advances in Neural Information Processing Systems 30, 5574–5584, Long Beach, CA, USA. 4-9 Decembber.

Kendall A., Grimes M., Cipolla R. (2015) ,“Posenet: A convolutional network for real-time 6-dof camera relocalization” ,In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), 2938–2946, Washington, DC, USA. 21-28 November.

Kingma D. P. Ba J. (2015) ,“Adam: A method for stochastic optimization” ,In Bengio Y. LeCun Y., editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA. 7-9 May.

Kneip L., Scaramuzza D., Siegwart R. (2011) ,“A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation” ,In CVPR 2011, 2969–2976, Colorado Springs, CO, USA. 20-25.

Laskar Z., Melekhov I., Kalia S., Kannala J. (2017) ,“Camera relocalization by computing pairwise relative poses using convolutional neural network” ,In The IEEE International Conference on Computer Vision (ICCV) Workshops, Venice, Italy. 22-29 October.

Lepetit V. Fua P. (2005) ,“Monocular model-based 3d tracking of rigid objects” ,Found. Trends. Comput. Graph. Vis., 1:1–89.

Lepetit V., Moreno-Noguer F., Fua P. (2009) ,“Epnp: An accurate o(n) solution to the pnp problem” ,International Journal of Computer Vision, 81(155):502–511.

Li C., Bai J., Hager G. D. (2018) ,“A unified framework for multi-view multi-class object pose estimation” ,In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 11220 LNCS, 263–281. Springer Verlag.

Li Y., Wang G., Ji X., Xiang Y., Fox D. (2020) ,“Deepim: Deep iterative matching for 6d pose estimation” ,International Journal of Computer Vision, 128:657–678.

Lin T.-Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollár P., Zitnick C. L. (2014) ,“Microsoft coco: Common objects in context” ,In Computer Vision – ECCV 2014, 740–755, Zurich, Switzerland. 6-12 September.

Lowe D. G. . (2004) ,“Distinctive image features from scale-invariant keypoints” , International Journal of Computer Vision, 60(2):91–110.

Lowe D. G. (1999) ,“Object recognition from local scale-invariant features” ,In Proceedings of the International Conference on Computer Vision, ICCV ’99, 1150–1158, Corfu, Greece. 20-27 September.

Martinez M., Collet A., Srinivasa S. S. (2010) ,“Moped: A scalable and low latency object recognition and pose estimation system” ,In Anchorage A. K., editor, 2010 IEEE International Conference on Robotics and Automation, 2043–2049.

Matthews A. G. G., Wilk M. V. D., Nickson T., Fujii K., Boukouvalas A., León-Villagrá P., Ghahramani Z., Hensman J. (2017) ,“Gpflow: A gaussian process library using tensorflow” ,Journal of Machine Learning Research.

Melekhov I., Ylioinas J., Kannala J., Rahtu E. (2017) ,“Relative camera pose estimation using convolutional neural networks” ,In Advanced Concepts for Intelligent Vision Systems, 675–687, Cham. Springer International Publishing.

Oberweger M., Rad M., Lepetit V. (2018) ,“Making deep heatmaps robust to partial occlusions for 3d object pose estimation” ,In Computer Vision – ECCV 2018, 125–141, Cham. Springer International Publishing.

Peng S., Liu Y., Huang Q., Zhou X., Bao H. (2019) ,“Pvnet: Pixel-wise voting network for 6dof pose estimation” ,In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 4556–4565. IEEE Computer Society, 16-19 June.

Persson M. Nordberg (2018) ,“Lambda twist: An accurate fast robust perspective three point (p3p) solver” ,In Proceedings of the European Conference on Computer Vision (ECCV), 312–322, Munich, Germany. 8 - 14 September.

Plantinga W., Dyer H., R. C. (1986) ,“Algorithm for constructing the aspect graph” ,In Annual Symposium on Foundations of Computer Science, 126–131, Cambridge, MA,

USA. 27-29 October.

Pong H. K. Cham T. J. (2006) ,“Object detection using a cascade of 3d models” ,In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 3852 LNCS, 284–293.

Rad M. Lepetit V. (2017) ,“Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth” ,In Proceedings of the IEEE International Conference on Computer Vision, 3848–3856, Venice, Italy. Institute of Electrical and Electronics Engineers Inc., 22-29 October.

Redmon J., Divvala S., Girshick R., Farhadi A. (2016) ,“You only look once: Unified, real-time object detection” ,In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779–788, Las Vegas, Nevada. 26-30 June.

Rublee E., Rabaud V., Konolige K., Bradskiv G. (2011) ,“Orb: An efficient alternative to sift or surf” ,In International Conference on Computer Vision, 2564–2571, Barcelona, Spain. 6-13 November.

Sandler M., Howard A., Zhu M., Zhmoginov A., Chen L. C. (2018) ,“Mobilenetv2: Inverted residuals and linear bottlenecks” ,In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, USA. 18-22 Jun.

Selvaraju R. R., Cogswell M., Das A., Vedantam R., Parikh D., Batra D. (2017) , “Grad-cam: Visual explanations from deep networks via gradient-based localization” , In 2017 IEEE International Conference on Computer Vision (ICCV), 618–626, Venice-Italy. 22-29 October.

Shotton J., Glocker B., Zach C., Izadi S., Criminisi A., Fitzgibbon A. (2013) ,“Scene coordinate regression forests for camera relocalization in rgb-d images” ,In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, CVPR ’13, 2930–2937, Washington, DC, USA. 12-18 October.

Shrikumar A., Greenside P., Kundaje A. (2017) ,“Learning important features through propagating activation differences” ,In Precup D. Teh Y. W., editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, 3145–3153, Sydney, NSW, Australia. 06-11 August.

Simonyan K., Vedaldi A., Zisserman A. (2013) ,“Deep inside convolutional networks: Visualising image classification models and saliency maps”.

Sock J., Kim K. I., Sahin C., Kim T.-K. (2018) ,“Multi-task deep networks for depth-based 6d object pose and joint registration in crowd scenarios” ,ArXiv, abs/1806.03891.

Soviany P., Ionescu R. T., Rota P., Sebe N. (2021) ,“Curriculum learning: A survey”.

Tekin B., Sinha S. N., Fua P. (2018) ,“Real-time seamless single shot 6d object pose prediction” ,In 10636919, Salt Lake City, Utah, USA. 18-22 Jun.

Terzakis G. Lourakis M. (2020) ,“A consistently fast and globally optimal solution to the perspective-n-point problem” ,In Vedaldi A., Bischof H., Brox T., Frahm J.-M., editors, Computer Vision – ECCV 2020, 478–494, Zurich, Switzerland. September 6-12.

Walch F., Hazirbas C., Leal-Taixe L., Sattler T., Hilsenbeck S., Cremers D. (2017) , “Image-based localization using lstms for structured feature correlation” ,In The IEEE International Conference on Computer Vision (ICCV), Venice, Italy. 22-29 October.

Xiang Y., Schmidt T., Narayanan V., Fox D. (2018) ,“Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes” ,In Robotics: Science and Systems Foundation. 8.

Xu B., Zhang L., Mao Z., Wang Q., Xie H., Zhang Y. (2020) ,“Curriculum learning for natural language understanding” ,In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 6095–6104, Online. 2-4 July.

Yu F. Koltun V. . (2016) ,“Multi-scale context aggregation by dilated convolutions. 4th international conference on learning representations” ,In 4th International Conference on Learning Representations, ICLR, San Juan, Puerto Rico. 2-4 May.

Zakharov S., Shugurov I., Ilic S. (2019) ,“Dpod: 6d pose object detector and refiner” , In Proceedings of the IEEE International Conference on Computer Vision, 1941–1950, Seoul, North Korea. 27 Oct– 2 Nov.

Zhou B., Khosla A., A. L., Oliva A., Torralba A. (2016) ,“Learning deep features for discriminative localization.” ,In The IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR), Las Vegas, Nevada. 26-30 June.

Zhou B., Lapedriza A., Khosla A., Oliva A., Torralba A. (2018) ,“Places: A 10 million image database for scene recognition” ,IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(6):1452–1464.

Zhu M., Derpanis K. G., Yang Y., Brahmbhatt S., Zhang M., Phillips C., Lecce M., Daniilidis K. (2014) , “Single image 3d object detection and pose estimation for grasping” , In 2014 IEEE International Conference on Robotics and Automation (ICRA), 3936–3943, Hong Kong, China. 1-7 June.

BIOGRAPHY

Muhammet Ali Dede received his B.Sc. from İzmir Institute of Technology in 2007 and his M.Sc. degree from Boğaziçi University in Computer Engineering in 2009. He worked as a software engineer, entrepreneur, and research assistant throughout his life. He is currently Leading ML Engineering Team in AKBANK. His research interests include anything computable yet he also spends time with uncomputable ideas.

APPENDICES

Appendix A: Publications within the Scope of the Thesis Study Dede, M. A. Genç, Y. (2022). Direct pose estimation from RGB images using 3D objects . Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi , 28 (2) , 277-285 Dede, M. A., Genc, Y. (2022). Object aspect classification and 6DoF pose estimation. Image and Vision Computing, 124, 104495.

Curriculum learning for object pose estimation. Submitted to in Image and Vision Computing (2022).