Australian National University

Creating a New Dataset for Efficient Transfer Learning for 6D Pose Estimation

October 26, 2018

Jack Henderson

u5561978

ENGN4712 - 12 Unit R&D Project

Supervised by

Professor Richard Hartley

of the

College of Engineering & Computer Science

Australian National University

Abstract

Current state-of-the-art pose estimation techniques rely on neural network models to accurately detect, localise and estimate the pose of target objects in a scene. One of the primary weaknesses in these methods is the difficulty in detecting new target objects which the network has not been trained on. We propose a new type of neural network structure which allows for efficient transfer of learning from the training set onto unseen objects. We identify a lack of existing datasets to suit this type of structure. Based on this, we formalise a method for a semi-automated process of camera calibration, hand-eye calibration and capturing of images for the object pose library. We also consider methods of augmenting the pose library to create a corresponding training set for the target objects. We demonstrate how to isolate the target object from the background, and a method of artificially rotating the camera viewpoint. Overall, we provide a foundational process for creating a new type of dataset, enabling the use of a new structure of neural networks for pose estimation. We further identify a number of key areas for further development of this dataset.

Source Code

All source code used in this project is available at
`https://bitbucket.org/jackhenderson101/engn4712-code/src`

Acknowledgements

I would like to take this opportunity to thank those who have provided me with support and guidance throughout the course of this project. In particular, my primary supervisor, Richard Hartley, has been instrumental to the direction and success of this project. I would also like to thank Viorela Ila, Alex Martin, and Kartik Gupta for their technical assistance.

Contents

1 Introduction

In the field of computer vision, pose estimation remains at the forefront of current research. The task involves estimating the translation and rotation of rigid objects relative to the reference frame of the camera. While current state-of-the-art techniques achieve very good performance on several challenging test datasets, there are still a number of areas where these approaches fall short. Handling occlusion and appearance changes has been the primary focus of recent research, however one factor which is severely limiting widespread adoption of these algorithms is their ability to easily and efficiently adapt to new, unseen objects.

Current leading approaches to pose estimation utilise machine learning and neural networks to perform the localisation of the target object within the scene, and provide a coarse estimate of the object's pose. Typically, this estimate is then refined using different heuristics to produce a more accurate pose. The issue with this approach is that the network needs to be trained on thousands, or tens of thousands of training images, showing different examples of the target object in various poses. Adding in a new, previously unseen target object is a non-trivial exercise, often requiring re-structuring of the network and re-training using a new training set of images.

We propose a new generalised type of network structure, involving the use of an object pose library as a parallel input to the network, alongside the query image. Consequently, in order to create and test networks of this structure, we require a new type of dataset with different properties to what is currently available in the literature. Thus, the focus of this report is on analysing what properties this type of dataset should have, and constructing a semi-automated process in which to capture the data required.

2 Background

In the field of computer vision, pose estimation is the task of identifying the position and orientation of a target object in a scene. We restrict the target objects to only rigid-body objects as a simple representation for pose cannot be defined for non-rigid objects. An object's pose is typically represented as 6-dimensional vector, representing the x,y,z spatial coordinates of the object and the rotation of the object. To differentiate the field from human pose estimation, the task can also be referred to as *6-D pose estimation* to explicitly refer to estimation of the 6 parameters defining both the position and orientation. In this

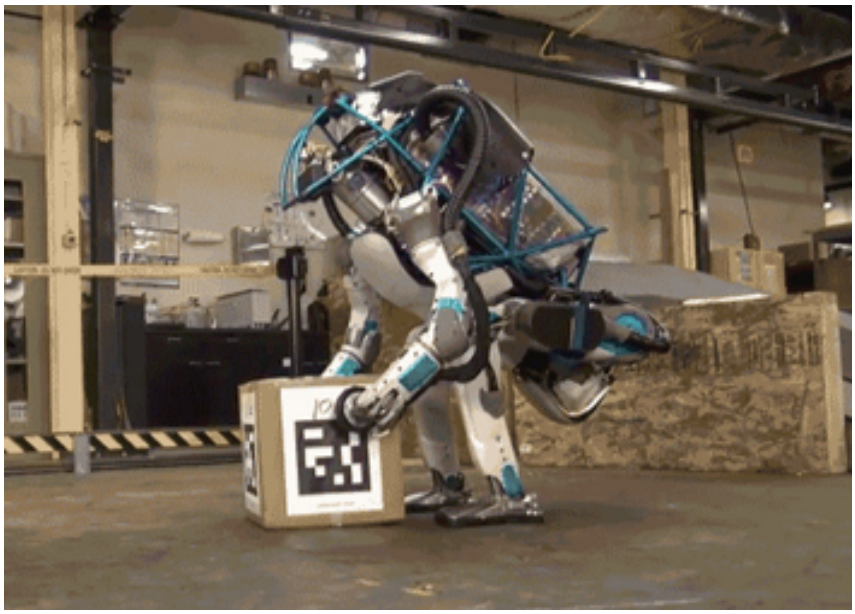


Figure 1: Example of a robot performing an object manipulation task and using fiducial markers to determine the object pose [1]

report, we also refer to *visual pose estimation* in relation to pose estimation algorithms which solely rely on visual information received from an RGB(/D) camera without any additional object markers or sensors.

Rotation can be represented using 3 values by axis-angle coordinates, Euler angles (roll, pitch, yaw) or Euler-Rodrigues parameters. Alternatively, to avoid any ambiguity in the order in which to apply the rotation and translation, we can represent the entire translation as a 4×4 homogeneous transformation matrix, which is the convention we use in this report. The pose of the target object is given with respect to the camera pose. If the camera pose is known with respect to the world reference frame and the camera is calibrated, then the target object's pose in the world reference frame can also be determined. It is also important to note that the target object must be known a-priori and its coordinate system must be well-defined.

2.1 Current and Potential Applications

One potential application of visual pose estimation is when it is used in conjunction with robotic manipulator to pick up and move objects, also known as 'Pick-and-Place' tasks. A example of a robot performing this task is shown in Figure 1. There are a number of fields in which this kind of task is required, from industrial to consumer level. The Amazon Picking Challenge [2] is an example of this type of task, where a robot must

identify objects in a cluttered box, use a gripper to pick up the object, and stow it into a different box. This has many applications in a number of difference warehouse scenarios, with Amazon's own fulfilment centres being the primary target. One of the key elements to performing well in this task is having an accurate estimate for the pose of the object that the robot is trying to pick up. In an environment such as an Amazon warehouse, where many thousands of products are stored, it would be impractical to place identifying markings or calibration patterns onto the objects, and thus visual pose estimation is suited well to the task.

The winning team of the 2017 Amazon picking challenge [3] used a rudimentary form of pose estimation in order to determine where the gripper should position itself to pick up the object. After segmenting the target object from the background, they perform Principal Component Analysis (PCA) to determine the principal axis of the object. The gripper is then aligned with this axis in order to stow the objects efficiently. Such a technique is not a true 6-D pose estimation algorithm, but provides sufficient information in order to achieve the desired task. They do note that this approach does not work well then the object is significantly occluded and that a more robust pose estimation algorithm would be beneficial.

Another field that utilises pose estimation is augmented reality (AR). The concept of augmented reality is about superimposing virtual objects into real-world scenes. In order to accurately position and project the virtual object into the scene, the pose of the camera and the objects in the scene must be known to a high degree of accuracy. AR is typically performed in a live on-the-fly setting, and so it requires pose estimation to be fast and efficient but also robust and continuous.

2.2 Difficulties and Challenges

The type of cameras used for pose estimation can vary, including stereo cameras, RGB-D cameras and standard RGB cameras. Both stereo and RGB-D cameras provide additional information about the depth of objects in the scene, which can greatly help to improve accuracy of pose estimation. One of the major challenges that affects monocular cameras is the limited amount of depth information that can be extracted. Identical objects at different depths exhibit both perspective changes and changes in apparent size. The further away objects are from the camera, the smaller these effects become which increases uncertainty.

Further to this, a number of other challenges make pose estimation more difficult, regardless of what type of camera is used. Occlusion, where the target object is partially obscured, can be particularly detrimental to some algorithms. Different lighting conditions, non-textured object surfaces, axes of object symmetry, reflections are other factors that need to be considered when designing algorithms to ensure that they are robust against these effects.

2.3 Metrics and Measurements of Success

When comparing different pose estimation algorithms, it is important to have metrics and measures by which to evaluate the success or failure of the algorithms in different scenarios. A number of different factors are considered desirable in a pose estimation algorithm. The most important of these is pose accuracy — i.e. how close is the pose estimated by the algorithm to the actual, ground-truth pose of the object. Beyond this, robustness and consistency are also important factors. In the case of pose estimation in videos, it is vital that the pose estimate remains consistent frame-to-frame. The estimate should also be robust to the challenging factors discussed above such as occlusion, reflections, and changes in lighting conditions.

Some commonly used metrics include 2D projection error, and the '5cm-5°' metric [4]. The 2D projection error uses the projection of the object's vertices into the image plane. The error is the average distance in pixels between the projection of the ground truth compared with the projection of the estimate. The 5cm-5° metric is a binary metric that considers an estimated pose 'correct' if it is within a 5cm translation and a 5° rotation of the ground truth. Such a metric is useful when the intended application does not require highly accurate pose estimations and robustness and consistency is a higher priority.

If a 3-D textured mesh model of the target object is available, the leading measure that is currently used is the Visible Surface Discrepancy (VSD) [5] measure. The model is rendered in both the ground truth pose and the estimated pose and then these renderings are compared to determine how visually similar they are, accounting for occluded regions. This metric has a number of useful properties, but they must be considered in the context of the intended application of the given algorithm. As this metric compares visual similarity, it may be possible for the pose estimate to be significantly different to the ground truth, and yet be rendered in a visually similar way. Whether this is a desired effect or not depends on the application. It does simplify a number of issues. If an object has one or more axes of symmetry, then it does not matter which instance is selected as they will be

rendered identically. However, for objects that are partially symmetric, the VSD score may be very low despite the estimated pose being significantly different from the correct value.

3 Literature Review

Pose estimation has been a long-studied field of computer vision and many different approaches have been taken in an attempt to create the best performing and most robust algorithm. Broadly, these approaches can be divided into three main classes of algorithms; marker-based, template matching, and learning-based. We explore these approaches and examine the advantages and drawbacks of each approach.

3.1 Fiducial Marker-based Pose Estimation

A common approach in many pose-estimation tasks is to use fiducial markers in the scene to provide additional information about the relative poses between the camera and these markers. Fiducial markers are artificially generated patterns which are placed into the scene that allow for efficient and accurate estimation of the relative pose between the camera and the marker. If the marker is then attached to an object of interest, then it is possible to determine the pose of the object relative to the camera. Additional markers added to scene can then be used to determine the objects pose relative to the world frame. An example of a fiducial marker in use in a robotics context is shown in Figure 1.

A popular implementation of the fiducial marker system is the ARTag [6]. An ARTag is a 2-D barcode-like pattern with a grid of 6×6 black or white squares. Each tag in a scene can have a unique non-symmetric pattern and thus will be uniquely identifiable. The corners of the marker can be identified in the image, and a resectioning, or perspective n-point (PnP), algorithm can be used to determine to pose of the marker.

Fiducial markers have a number of advantages when used for pose estimation. They have very low false positive rates - ARTag claims the probability of a false positive is below 0.004%, even with multiple markers in the scene [6]. This is extremely valuable in applications where high levels of robustness are required. The processing required to identify markers in a scene and estimate their pose is simple and efficient, allowing for real-time pose estimation – an essential element of augmented reality systems. Also, because the markers are black and white they have very high contrast which makes them robust to a wide range of lighting conditions.

However, despite their numerous advantages and popularity, fiducial markers have a number of limiting factors. The biggest issue is that they have to be manually attached to objects that require tracking. While this may be practical for small-scale experimental

setups in controlled conditions, it does not lend itself well to real-world scenarios. For example, if a fiducial marker system was used for the Amazon picking challenge, it would effectively require markers to be added to every single product in Amazon's inventory — a very unlikely eventuality. Some objects may be too small to add markers, or they may be of an unusual shape, texture or material making attaching markers difficult. Occlusion is another area in which fiducial markers perform poorly. If a tag is even partially occluded, the performance degrades significantly. For ARTag, occlusion level over 4% are enough to reduce the rate of correct recognition below 100% and at levels above 14%, recognition rates drop to zero [7]. Other markers perform similarly. However, it's important to note that when a marker is occluded it's much more likely that the tag won't be recognised at all rather than produce an incorrect estimate of the pose. This issue can be resolved by adding additional, redundant markers to the target object, but again this is impractical on large scales.

3.2 Template-based Pose Estimation

Recognising that marker-based approaches are not suitable for a wide range of tasks, effort has been made towards pure visual-based pose estimation that does not require additional markers. The most prominent and best performing algorithm in this class is the LINEMOD algorithm presented by Hinterstoisser et al. [8] and its derivatives [9]. The algorithm uses both the RGB image and a depth map of the scene to extract a set of features for the image. These features are then matched against a set of templates, which are generated from rendering a 3D model of the object in different poses. Because this process involves template matching, the runtime of the algorithm is dependent on the number of templates used, but in order to get more precise results a large number of templates is needed. This results in a comparatively slow process for pose estimation. Also, due to the requirements for both RGBD images, and 3D models of the target objects to generate the templates, the areas in which this algorithm is applicable are more limited. It appears that research into this class of algorithms has diminished in favour of machine learning approaches.

3.3 Visual Pose Estimation using Machine Learning

With the arrival of modern machine learning techniques and neural networks, there have been a number of developments in relation to applying these techniques for visual pose estimation. Rather than having to rely on fiducial markers, approaches using machine

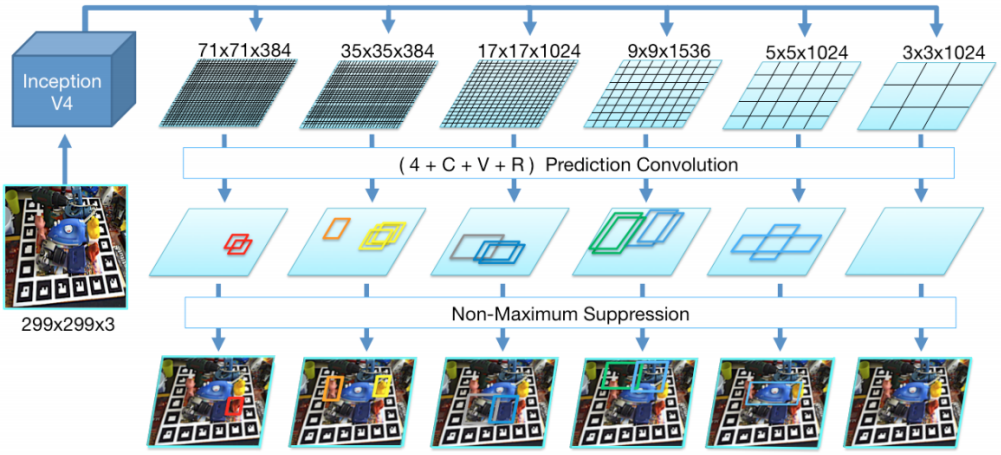


Figure 2: Structure of the SSD-6D network. C denotes the number of object classes, V the number of viewpoints and R the number of in-plane rotation classes. [10]

learning aim to estimate pose directly from the visual representation of the object without any additional markers or sensors. The goal of machine learning based approaches is to recognise the pose of arbitrary objects with no artificial markings and ideally be robust to moderate to high levels of occlusion.

The BB8 algorithm, proposed by Rad and Lepetit [4], is an example of one way to apply machine learning approaches. In this method, the pose estimation is a two-step process. Firstly, the object is localised in the 2D image use a simple image segmentation network, trained to segment foreground from background. Once the target has been localised in the image, a fixed-size window is extracted from the image centred on the object. This window is then used as the input for a secondary network, which predicts the 8 corner points of a bounding cuboid around the object. These predicted points are then used in a constrained minimisation problem to estimate the true pose of the target object. The pose can optionally be further refined if a 3D model of the target object is available. The 3D model of the target object is rendered in the estimated pose and compared to the view of the object in the image. The pose is then refined to minimise the difference between the rendered view and the actual view of the target object.

The main issue with the BB8 algorithm is that it can only be trained on a single object at a time. When estimating the pose of a new object, the entire network must be retrained. Also, as the algorithm involves multiple stages, it is not trainable end-to-end — both the object localisation stage and the corner-point estimation stage must be trained separately.

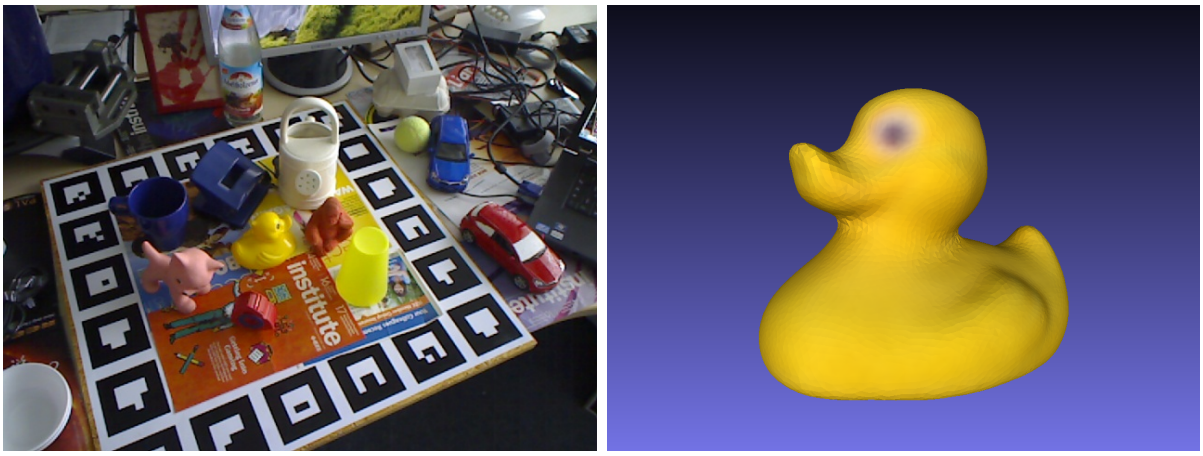
A different approach that is end-to-end trainable is the SSD-6D network proposed by

Kehl et al. [10]. It builds on a pre-existing object detection framework, the Single Shot Detection (SSD) network [11]. The original SSD network is able to localise target objects within an image using what is essentially a classifier-style network, and classifying between a number of different rectangular region proposals. The SSD-6D network builds on the same structure, changing the initial classification layers from VGG-16 to the InceptionV4 [12] network. The output of the network is also modified to perform further classification tasks; for each of the region proposals in the output, the network also classifies the type of the object in the scene, which viewpoint the object is being view from, and also the in-plane rotation of the camera with respect to the object. Non-maximum suppression is used to select the best region proposal, and then the classification values for this region are used to reconstruct the 6D pose. A diagram of this network structure is shown in Figure 2. However, because the network is classifying the target object into a discrete set of viewpoints, a further refinement step must be performed to generate accurate results. Similar to BB8 [4], this involves rendering a 3D model of the object in a small range of poses.

Tekin et al. [13] present a different network structure, which is based on the YOLO object detection network. In contrast to most other approaches including both BB8 and SSD-6D, it does not require the final pose refinement stage. This means that the algorithm does not require the use of the 3D model of the target object and allows it to be used on a much broader range of datasets. Similar to the BB8 algorithm, the network directly predicts the projections of the corner points of the bounding cuboid of the object, and then uses a PnP algorithm to solve for the object pose. Results from the network are mixed; compared to SSD-6D the performance is generally worse, however the computational performance is significantly faster. The main reason for the drop in performance is due to the lack of the pose refinement step. If the results are considered against SSD-6D without pose refinement, then results are markedly better.

3.4 Datasets

A number of different datasets have been created for work in this field. At the most basic level, a dataset should consist of a set of images of a target object, and a set of associated ground-truth labels identifying the transformation between the camera frame and the target object frame. As the target object will typically not have an intrinsic coordinate system associated with it, the coordinate system is defined by the ground-truth values. As the performance and robustness of pose estimations algorithms improves, new datasets have been developed to be more challenging and complex, focussing on properties that



(a) Scene containing target object *(b) 3D Model of target object*

Figure 3: Example data from the Hinterstoisser dataset [9]

expose weaknesses in existing algorithms.

The biggest challenge in creating a dataset for 6D pose estimation is in determining the exact value for the transformation between the camera and the target object. We examine a selection of different existing datasets and explore the different methods in which ground truth values are determined.

3.4.1 Hinterstoisser Dataset

Hinterstoisser et al.'s dataset [9] is also known as the LINEMOD dataset for its association with the LINEMOD algorithm proposed in the same paper. The dataset includes 15 different household objects with over 1000 images of each object. Each object is rigid, mostly textureless and can be discriminated based on shape and colour. The images include highly cluttered scenes on a table with the target object occasionally occluded by another object. Ground truth poses are provided as a rotation and translation matrix, representing the relative pose between the camera and the target object. An example image from the Hinterstoisser dataset is shown in Figure 3.

The dataset was captured using a Kinect RGBD camera, and thus both RGB images and depth maps are included. It is assumed that the camera was hand-held when capturing the dataset as there appears to be no consistent pose trajectory. However, poses have been selected to ensure a roughly even distribution of poses across the hemisphere centred on the target object at a range of distances. In order to determine the pose of the camera relative to the world, fiducial markers have been placed on the perimeter of the table. The object was affixed to the centre of the table, giving a known transformation between

the world coordinates and the target object coordinate frame. Other objects are added randomly to the table to add clutter and provide for occlusions. Across the range of the dataset, these occluding objects are occasionally rearranged to add more variety to the images.

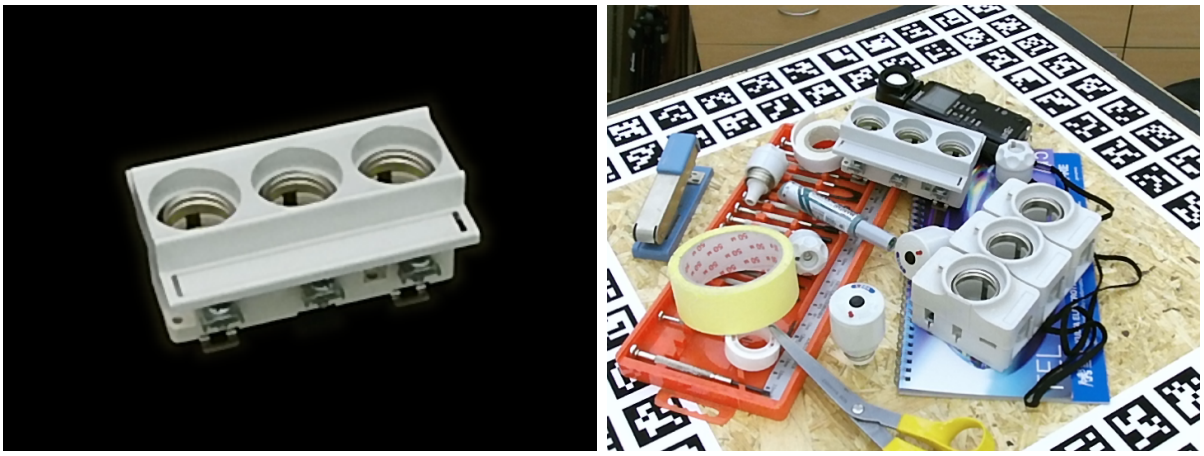
The Hinterstoisser dataset also includes 3D models for each of the 15 objects, both as a textured mesh and as a textured voxel representation. However, it's unclear exactly how these models were generated and whether there is any manual input into the process. These can be used to artificially generate images of the target object in any arbitrary pose. This technique is frequently used in machine learning algorithms to generate vast quantities of training images by rendering the object in different poses in a variety of scenes [4], [10], [14].

One issue with the use of fiducial markers for the dataset is that it renders the dataset unusable in the training set of neural networks. This is because there is the potential for the neural network to learn the appearance of the markers rather than the target object itself. Considering that the markers always remain fixed relative to the target object, it's highly probable that this would happen. Thus, unless the markers can somehow be removed, the training set for a neural network based on this data could only contain the artificially generated renderings of the 3D model discussed above.

3.4.2 T-LESS Dataset

Hodan et al. [15] present a more challenging dataset, T-LESS, which consists of approximately 50,000 images of 30 different consumer electrical components. The objects are made of white plastic and have very little texture, colour variation, and often have multiple axes of symmetry. This makes T-LESS a particularly challenging dataset for 6D pose estimation. The dataset is split into a training set, and a test set. The training set consists of individuals objects on a black background and the test set consists of 20 different static scenes with different levels of clutter, backgrounds and numbers of objects. Examples of both the training set and test set are shown in Figure 4.

The dataset was captured using both RGB and RGBD cameras in a fixed range of azimuths and elevations. To control azimuth, objects were placed on a movable turntable which was manually rotated in 5° increments. Elevation was controlled by placing the cameras in a jig, allowing the elevation to be manually adjusted in 10° increments. The views from the upper hemisphere are captured first, and then the object is flipped upside-down to capture the lower hemisphere resulting in a total of 1,926 training images for each



(a) Training set *(b) Test set*

Figure 4: Example data from the T-LESS dataset [15]

object. Two types of 3D Models of the dataset are also provided, one that is manually created and another semi-automatically generated from the training dataset.

Fiducial markers are used on the perimeter of the turntable to determine an accurate value of the target object pose, but these are cropped or masked out in the training set, eliminating the issue seen in the Hinterstoisser dataset. While these fiducial markers provide an accurate estimate of the camera-to-world transformation, they provide no information about the world-to-object transformation. This is resolved by using the 3D models of each object and manually aligning a rendered version of the model to the image. Once aligned, the world-object transformation can be used for all images of the same scene.

Overall, the techniques used to create the T-LESS dataset result in a high-quality images with accurate poses for a wide set of challenging objects. However, some parts of the method are extremely time-intensive and require a high level of manual adjustment and intervention to create the dataset. While the samping method captures a dense range of elevations and azimuths, the images are all captured at a fixed distance to the target object and the principal axis of the camera is aligned with the axis of rotation of the turntable.

3.4.3 Rutgers APC Dataset

The Rutgers APC dataset [16] was created to emulate the types of objects and scenes that are used in the Amazon Picking Challenge [2]. It contains a set of 24 different objects, some non-rigid, some transparent, arranged on a set of shelves/bins. The objects are

(a) Scene containing the target object *(b) 3D Model of target object*

Figure 5: Example data from the Rutgers APC dataset [16]

placed each of the different shelves and surrounded by a variable number of other objects to add clutter. Three different views of each scene are provided: one directly front-on to the shelf, one slightly to the left and one the same distance to the right. An example image from the Rutgers APC dataset is shown in Figure 5.

The dataset is captured using a Kinect RGB-D camera mounted to the end of a robotic arm. The robotic arm is positioned relative to the shelf to capture each image. This provides the camera-to-world transformation but, again, the world-to-object transformation is unknown. Similarly to the T-LESS dataset, this is performed by manually aligning a rendering of a 3D model of the object with the image.

Because of the nature of this dataset, namely that the number of different poses observed is very small and the distribution of these poses is not consistent, using it as a training set is not practical. However, it does have value as a test set. Again, this means that the training set must be created by artificially rendering the provided 3D models of the objects into whatever poses are required. Also, while the capture of images has been automated through the use of a robotic arm, manual intervention is still required to determine ground truth poses and the world-to-object transformation.

4 Project Scope

As discussed in Section 2.1, pose estimation can be used in warehouse automation as part of a pick-and-place robotic system or as part of an augmented reality system. In both of these scenarios, it is easy to imagine a situation where a neural-network-based pose estimation algorithm has been trained on a known set of target objects and then, at some point in the future, a new object is required to be added to the set of target objects. For example, in a warehouse system, a new product could be added to the inventory of the warehouse. Minimising both the amount of computational effort and human effort required in this process would be a highly valuable from the user's perspective. However, from our analysis of current pose estimation techniques we conclude that this would be a labour-intensive and time-consuming process.

Current implementations of neural network algorithms for pose estimation require extensive training on large corpus of images for each of the target objects. Thus, if we wish to estimate the pose of a new, unseen object, we must: capture a collection of annotated images of the new object in a range of poses, modify the structure of the network to incorporate the additional object label, and finally retrain the entire neural network with the updated dataset. As discussed in Section 3.4, the methods in which the ground-truth-annotated datasets are created often involves human intervention to generate 3D models of the target object, align the rendered 3D models with images, or manually adjust the position of a camera or object to capture the full range of poses. Of the three different approaches to creating datasets we examine in Section 3.4, all require a high degree of human interaction to capture and annotate the dataset.

Once a dataset has been captured for the new target object, the neural network then needs to be retrained to be able to estimate poses. In most cases, including SSD-6D [10], the structure is designed to detect a fixed number of classes and thus the structure of the network will also need to be modified to accommodate detection of the new object. These networks also have very deep and complex structures, for example SSD-6D contains upwards of 50 layers. Retraining such a network requires significant levels of computing power and time, which is not ideal in a real-world scenario.

If we consider a new network structure, we could potentially avoid these issues and allow for new target objects to be added to the neural network much more efficiently and without the need for extensive retraining. Currently neural network approaches use only the query image as input to the network and generate the pose estimate from this information. Some algorithms also determine the class of object, but we restrict our

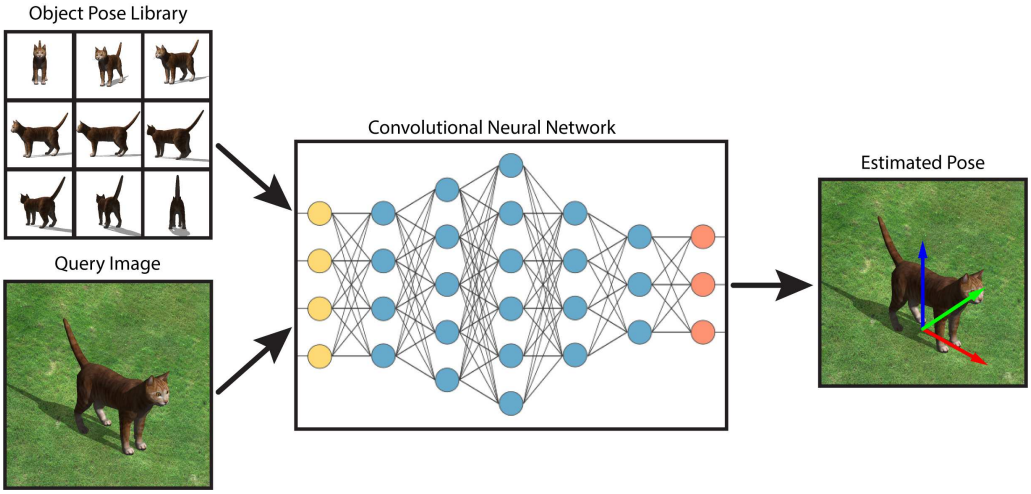


Figure 6: Proposed structure of neural network, adding in object pose library in parallel to query image.

approach to only consider object pose, given that the target object class is known a-priori.

One potential way to create a new class of neural networks would be to use a 'library' of target object poses as an additional input into the neural network alongside the query image. The object pose library would contain a finite set of reference images of the target object in a discrete range of poses. A diagram of this network structure is shown in Figure 6. Under this structure, the goal of the neural network would essentially be to match the query image to the closest image in the pose library and perform additional refinement in the case that the target object pose is not exactly coincident with a particular pose in the library.

We have not found any evidence in the literature that this type of network has been proposed previously, thus it seems a worthy avenue of exploration and experimentation. Whether this style of network will be effective, accurate or efficient remains to be seen. However, before we can attempt to construct or evaluate networks of this structure we first require an available dataset to use. The dataset should consist of a set of target objects, each with a corresponding pose library, and a set of query, or training, images. It should also be relatively simple to add new objects into this dataset by creating a new pose library for the object.

Thus, the aim of this project is to create a system which can generate a dataset for use in transfer learning-style pose estimation networks. This involves generating the object pose library as well as a set of realistic training images of objects that are in the library. For the reasons discussed above, the creation of the pose library should involve minimal human intervention.

5 Methodology

5.1 Overview

In the process of creating an object library and a training set, we require the camera to be placed in a specific set of poses relative to the target object. Due to the requirements of high accuracy and minimal human intervention, we use a robotic arm with the camera attached to the end effector. This allows us position the camera in the required poses and also determine the ground truth data for each of these poses.

We divide the process of creating the dataset into three separate parts;

1. **Calibration**
   Calibration is an essential part of the process to ensure the accuracy of the dataset is as high as possible. The calibration process involves calibrating the properties of the camera, as well as the robotic arm which the camera is attached to and their relationship between each of the key coordinate frames.

2. **Generating the object library**
   The object library should contain sufficient information so that the neural network can determine the pose of an object from the query no matter what pose it is in. Thus, we need to determine exactly how much data is sufficient for this purpose and propose a way to capture the required data.

3. **Generating training images**
   The training images should consist of images the target object in as wide a variety of poses as possible to ensure that the trained network is robust to different phenomena. Ideally, these images should include examples from a wide variety of poses, levels of occlusion, lighting conditions, and backgrounds. We should aim to create images that are as realistic as possible, in order to best represent the types of images that the neural network will be used in a real application.

5.2 Notation and Conventions

We define the following symbols and conventions:

- A right-hand coordinate system is used for all coordinate frames.

- The World frame, W, is fixed to the earth at an arbitrary point.
- The Robot Base frame, B, is fixed to the mounting platform of the UR5 robotic arm. The position and alignment of this frame is consistent with the representation used in the UR5 control software.
- The Robot Hand frame, H, is fixed to the Tool Centre Point (TCP) (a.k.a. the end-effector) on the UR5 robotic arm, consistent with the representation in the UR5 control software.
- The Camera frame, C, is fixed to the optical centre of the camera, which is mounted to the TCP of the UR5 robotic arm. The X-Y plane is aligned with the image sensor and the Z-axis points along the principal ray.
- The image frame, I, is a 2-dimensional frame aligned to the pixel grid of the image. The origin is in the top-left corner of the image. The x-axis along the horizontal edge positive to the right, and the y-axis is along the vertical edge of the image, positive downwards. A unit vector along either the x-axis or y-axis is equal in length to one pixel.
- P_X represents a point in 3-D space in relation to the reference frame X. It is defined in homogeneous coordinates, i.e:

$$P_X = w \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_X$$

  where w is an arbitrary non-zero scalar. If P is defined in the image frame, it only has three values:

$$P_I = w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_I$$

- $\mathbf{T}_{XY}$ represents a coordinate transformation from the reference frame Y to X. It is represented as a 4×4 homogeneous transformation matrix:

$$\mathbf{T}_{XY} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$$

  such that $P_X = \mathbf{T}_{XY} P_Y$, where $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{t}$ is a 3×1 translation vector.

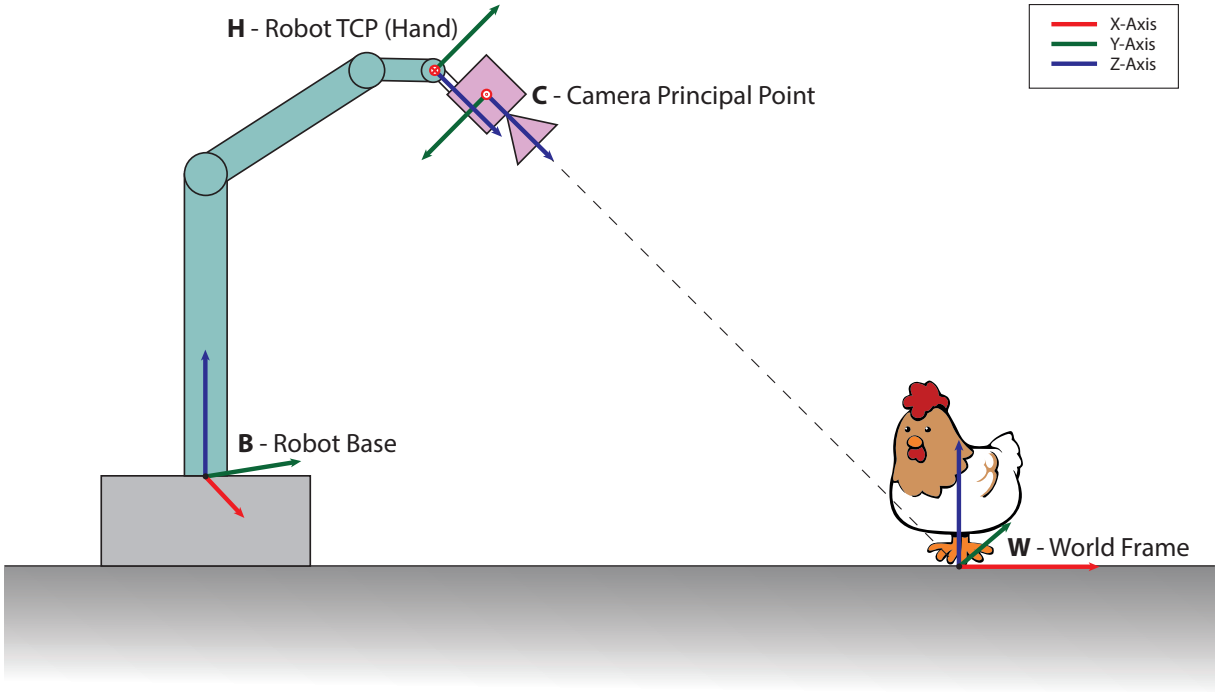


Figure 7: Diagram of experimental setup showing relationship between coordinate frames.

- K represents the camera intrinsics matrix, and is defined as

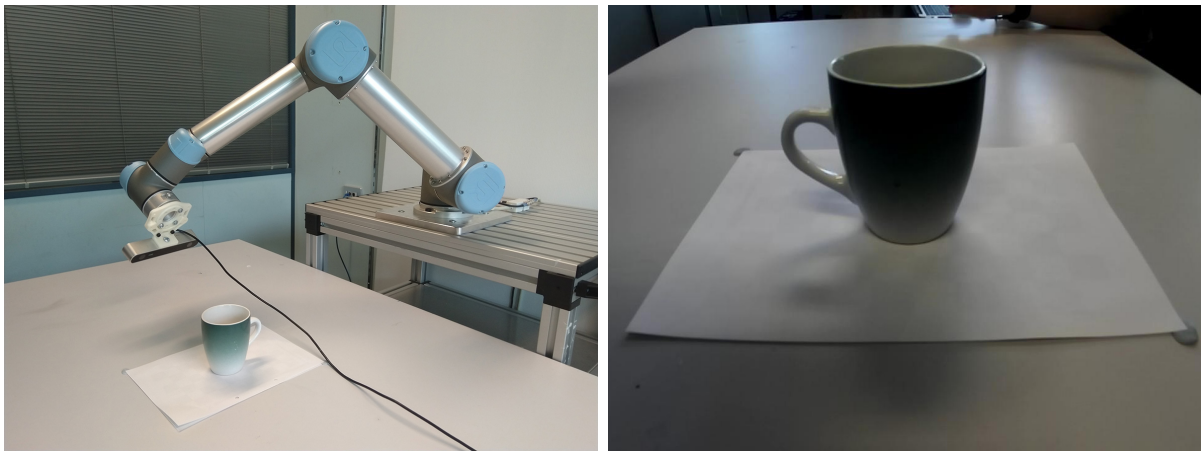
$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

  where f_x, f_y are the focal lengths in the x and y directions, and c_x, c_y are the principal point in the image. We do not consider skew.

5.3 Experimental Setup

In order to create a system to capture accurate and repeatable datasets, we utilise a Universal Robotics UR5 6-axis robotic arm. A camera is attached to the end-effector of the robot which allows us to place the camera in specific poses relative to the object of interest to a high degree of precision. An image of the setup is shown in Figure 8.

A set of tools were developed in MATLAB in order to control the UR5, the camera and perform all of the related image processing and pose generation. Three separate modules were developed to perform the following tasks;



(a) Example pose of robot for collecting data. (b) Corresponding image captured from camera.

Figure 8: Experimental setup for capturing target object images.

1. Camera calibration and Hand-Eye coordination
2. Target object image capture
3. Image post-processing and training set augmentation

The UR5 is connected via an Ethernet connection and commands are sent from MATLAB via TCP/IP. An accompanying script running on the robot control software interprets the commands and positions the robot in the commanded pose. Commanded positions specify the pose of the robot hand (TCP) with respect to the robot base.

The camera used is a StereoLabs ZED camera, which captures images at a resolution of 1280×720 pixels[1]. It is connected via USB 3.0 to the controlling computer. While the camera does have stereo capabilities, these are not utilised for the purpose of this project.

5.4 Camera Calibration and Hand-Eye Coordination

Before accurate images can be captured from the system, two properties must first be calibrated, the camera intrinsic parameters and the transformation between the robot end-effector and the optical centre of the camera.

The camera intrinsics include the focal length, principal point and skew factors. Additionally, distortion coefficients can be used to rectify optical distortion in the image.

[1]While the ZED camera is capable of capturing images at a resolution of 2208×1242, due to technical difficulties related to driver issues on Mac OS, we were limited to a resolution of 1280×720.

These values can be obtained through the use of MATLAB's inbuilt camera calibration toolbox. Firstly, a planar checkerboard pattern of known dimensions is placed in the centre of the workspace. The robot is then used to move the camera to a variety of different poses and an image of the checkerboard is taken at each pose. The camera calibration toolbox is then able to detect checkboard pattern in each image, and provide a value for the camera intrinsics parameters, including the distortion coefficients. We use 74 images of the checkerboard patterns from a range of different poses to determine the camera intrinsics. From this, we determine the camera intrinsic matrix to be:

$$\mathbf{K} = \begin{bmatrix} 703.13 & 0 & 645.58 \\ 0 & 701.49 & 383.41 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

which is consistent with the manufacturer-stated focal length of 700 px. Ideally, for the 1280×720 px image, the principal point should be at $(640, 360)$ which is different to the value measured from calibration. However, re-projecting known points back into the image coordinate system showed that the calibrated values were correct.

In addition to the camera intrinsics, the calibration toolbox also provides the relative poses between the camera and the checkerboard for each image. If we define the corner of the checkerboard to be the world origin, we then have a transformation, $\mathbf{T}_{C_iW}$, from the world-frame to the camera-frame for each pose, i, the camera was in. By recording the commanded position of the robot, we also know the transformation from the robot hand to the robot base, $\mathbf{T}_{BH_i}$, in each pose. There are still two remaining unknowns, both the transformation between the world-frame and the robot base, $\mathbf{T}_{WB}$, and the transformation between the robot hand and the camera optical centre, $\mathbf{T}_{HC}$. These transformations can be related in the following equation:

$$\mathbf{T}_{C_iW}^{-1} = \mathbf{T}_{WB} \times \mathbf{T}_{BH_i} \times \mathbf{T}_{HC}$$

Determining $\mathbf{T}_{HC}$ is a common robotics problem known as hand-eye calibration. To preform this step, we follow the well-established method proposed by Tsai and Lenz [17] and implemented into MATLAB by Lazarevic [18]. The resultant transformation between the hand and the camera is shown in Figure 9.

Once $\mathbf{T}_{HC}$ has been determined, the final remaining unknown is the transformation between the robot base and the world frame, $\mathbf{T}_{WB}$. While the world frame can be placed arbitrarily, it is useful to define a specific location so that objects can be placed in the world relative to a fixed point. From the camera calibration procedure, we have $\mathbf{T}_{C_iW}$ for

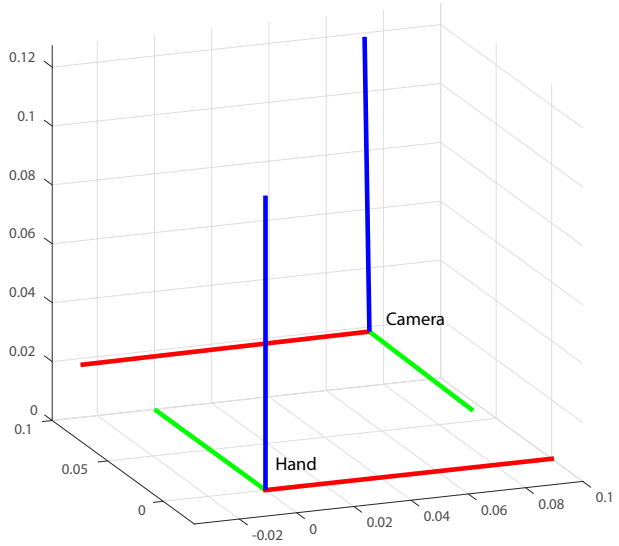


Figure 9: Calibrated transformation between the robot hand and the camera reference frames.

each of the i poses used for calibration. In this case, the world frame is defined by the calibration grid used, with the origin at the corner of the grid, and the frame aligned to the grid lines. The z-axis is perpendicular to the grid paper, and in our case it points down, through the table. In order to get an accurate value for $\mathbf{T}_{WB}$, we take the average across all the calibration poses:

$$\mathbf{T}_{WB} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{T}_{BH_i} \times \mathbf{T}_{HC} \times \mathbf{T}_{C_iW})^{-1} \tag{2}$$

However, taking a simple average does not ensure that the resulting transformation matrix represents only a rotation and translation. To constrain the transformation, we decompose it into the rotation and translation components:

$$\mathbf{T}_{WB} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{3}$$

For $\mathbf{R}$ to be a valid rotation matrix, it must be orthonormal. A simple way to ensure this is by taking the Singular Value Decomposition (SVD) of $\mathbf{R}$ and replacing the singular values with the identity matrix.

$$\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \tag{4}$$

$$\mathbf{R}' = \mathbf{U}\mathbf{I}\mathbf{V}^\top \tag{5}$$

While this approach is not necessarily the most optimal way to average rotation matrices,

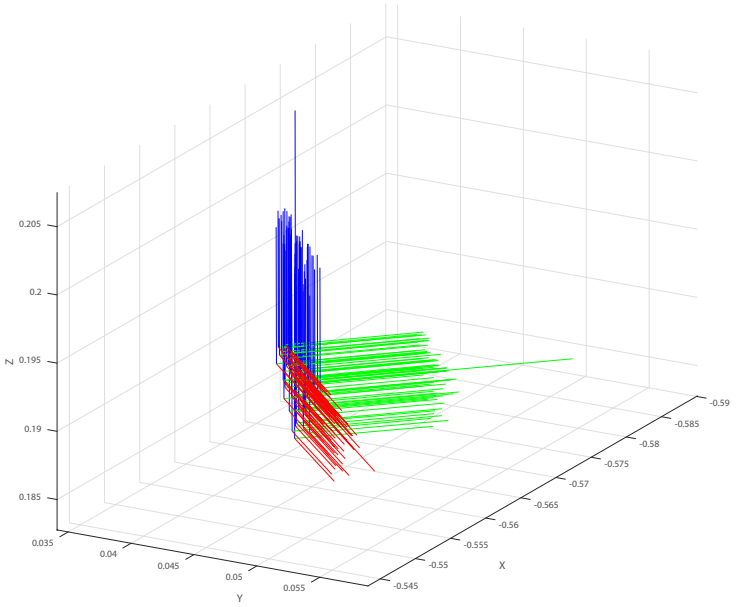


Figure 10: Robot base reference frame relative to world frame. Showing the result of each individual calibration pose $\mathbf{T}_{W_iB}$*, and the averaged transformation (longer axis lines),* $\mathbf{T}_{WB}$*, calculated in Eqn 2.*

it suffices for the purposes of this project. We show in Figure 10 that all of the calculated rotations are very similar, and the major variation is in the translation of the origin.

5.4.1 Image Distortion

The pinhole camera model is not sufficiently complex to capture all of the real-world effects when projecting an image onto a plane. One of the major factors that is not modelled is radial distortion, which distorts the image proportionally to the distance from the principal point. We model this distortion effect with the following equation:

$$\begin{bmatrix} x \\ y \end{bmatrix}' = (1 + k_1 r^2 + k_2 r^4) \begin{bmatrix} x \\ y \end{bmatrix}, \quad r = \sqrt{x^2 + y^2} \tag{6}$$

where k_1 and k_2 are the distortion coefficients, which are calibrated through the MATLAB camera calibration toolbox. In the above equation, (x, y) represent normalised, unit-less image coordinates, which are related to the image frame by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{x_I - c_x}{f_x} \\ \frac{y_I - c_y}{f_y} \end{bmatrix} \tag{7}$$

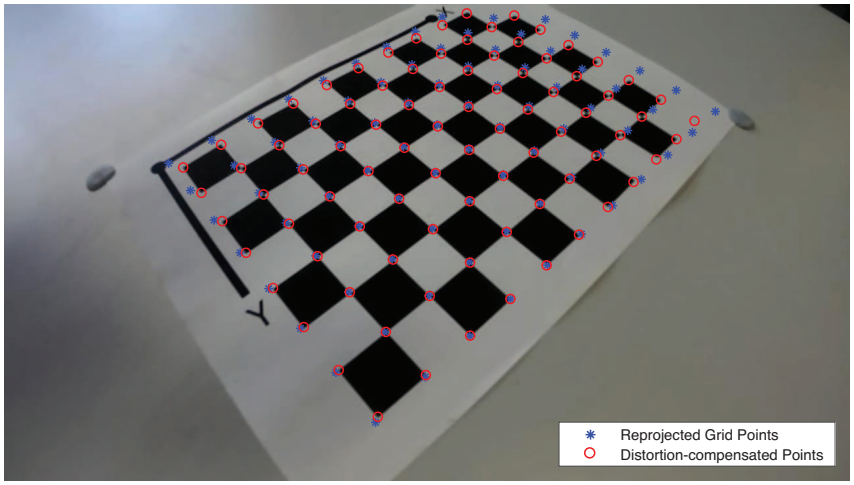


Figure 11: Example of the effect that radial distortion has on projecting world points into the image plane.

The radial distortion can be corrected for in two different ways. One way is to apply the distortion when projecting points into the image plane. An example of this, and a comparison to the non-distorted projection is shown in Figure 11. We observe that, for the camera we used, the distortion has a significant impact on points towards the edge of the frame, and without compensation, the projected points are significantly displaced from the true location. The alternative way to compensate for distortion is to warp the entire image with the inverse of the distortion. This means that points can be projected into the image plane using the standard pinhole model without having to compensate for distortion. This is especially useful for further image processing, which we discuss in later sections.

5.5 Generating Object Library Images

5.5.1 Choice of Poses for Library

When creating the object pose library, we must consider which poses to capture, and how this will affect the accuracy of the resulting neural network that is trained on this data. The pose library should aim to capture a wide enough range of poses to fully describe the target object, but be sufficiently small enough both to limit redundancy in poses and ensure the data can be captured in a timely manner.

To limit the pose space, we add several constraints; firstly, we only consider poses a fixed distance from the target object. While the appearance of an object may change

at different distances due to perspective effects, the effect is negligible at large enough distances. Thus it would be mostly redundant to capture multiple images at varying depths, when a single fixed depth would suffice. We also constrain the in-plane rotation of the camera such that the y-axis of the camera is always parallel to the x-y plane of the world frame. In other words, the camera is always level with the horizon. The final constraint is that the principal axis of the camera points towards the origin of the world frame. If the target object is then placed at the world origin, then all the images will have the target object in the centre of the frame.

The above constraints limit the available set of poses to the surface of a sphere centred on the world origin. However, there remains the choice of which points on the sphere to use. Choosing to sample a uniform distribution of azimuths and elevations, similar to how the T-LESS dataset [15] was constructed, would result in oversampling near the poles and comparative undersampling at low elevations. Instead, we use an algorithm presented by Deserno to sample equidistributed points on the surface of the sphere [19].

In an ideal world, we would be able to capture the entire distribution of poses of the sphere, however there are a number of implementation-specific issues that impose further limitations. Firstly, as the object is placed on a table, any pose in the lower hemisphere is unreachable. This could be overcome by capturing the upper hemisphere and then manually flipping the object upside-down to capture the lower hemisphere. Further to this, the workspace of the robot is limited by the length of the robot arm and manoeuvrability of its joints. This means that in some cases, the robot cannot reach the furthest points on the sphere and then point the camera towards the target object. We overcome this by separating the pose set into two halves and rotating the object 180° so that the robot can reach the required poses.

Even accounting for the above constraints, there are still a number of poses that are unreachable — these are due to the robot physically colliding with either itself, its supporting structure, or the table. Due to the complex nature of the trajectories of each joint of the robot and the geometry of the camera mounting hardware, it is intractable to predict which poses will cause collisions. We found that rotating the camera 180° about the camera z-axis helped to reduce the number of self-collisions, but did not eliminate them. It also meant that the images from the camera would be inverted. We also impose a minimum height for the camera to ensure the robot does not collide with the table.

Eventually, we determine a set of poses that satisfies all of the constraints and does not result in a collision during the full trajectory. We select 50 poses in one half of the upper hemisphere, at a fixed distance of 0.3 m from the target object, excluding poses less

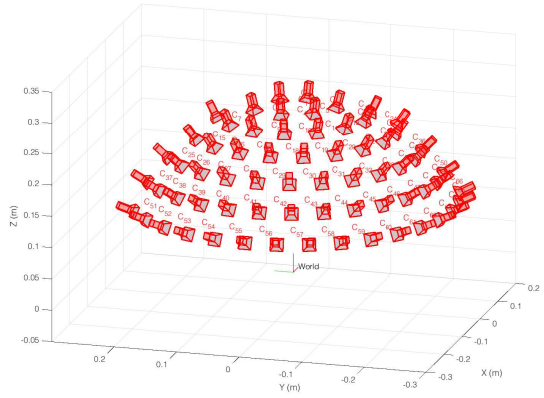


Figure 12: Camera pose for each image in the object library

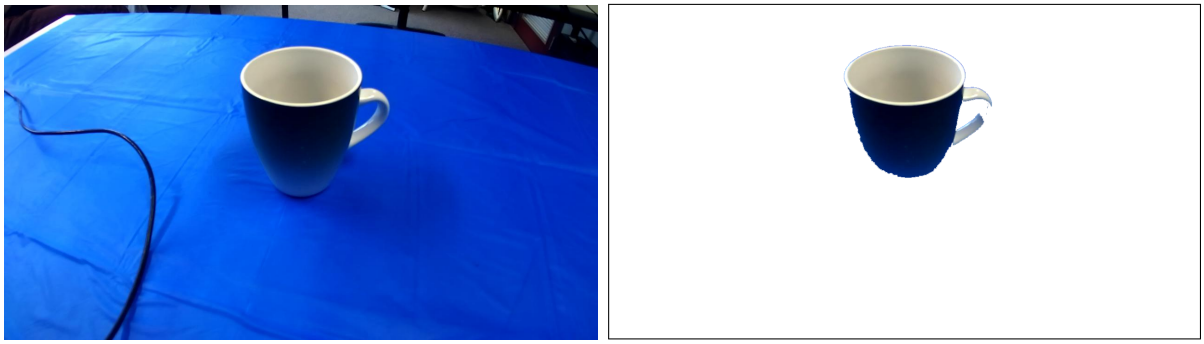
than 0.15 m above the table. A visualisation of these poses is shown in Figure 12.

The algorithm we use to generate a set of camera poses, and their corresponding robot hand poses is as follows:

1. Generate sample points, $P_i = [x, y, z]^\top$ using algorithm in [19]
2. Remove points that are out of reach of the robot arm
3. Determine required rotation to point camera z-axis towards the world origin and keep y-axis level with horizon:
   (a) Roll: $\alpha = \pi$ (to invert the camera)
   (b) Pitch: $\beta = -\tan^{-1}\left(\frac{z}{\sqrt{x^2+y^2}}\right)$
   (c) Yaw: $\gamma = \tan^{-1}\left(\frac{y}{x}\right)$
   (d) $\mathbf{R}_{WC_i} = \mathbf{R}_z(\gamma) \times \mathbf{R}_y(\beta) \times \mathbf{R}_x(\alpha) \times \mathbf{R}_y\left(\frac{\pi}{2}\right)$
4. $\mathbf{T}_{WC_i} = \begin{bmatrix} \mathbf{R}_{WC_i} & P_i \\ \mathbf{0} & 1 \end{bmatrix}$
5. $\mathbf{T}_{BH_i} = \mathbf{T}_{WB}^{-1} \times \mathbf{T}_{WC_i} \times \mathbf{T}_{HC}^{-1}$

5.5.2 Foreground Isolation

In order to create a useful object pose library, the target object (foreground) should be completely isolated from the background. The isolated target object will also be useful



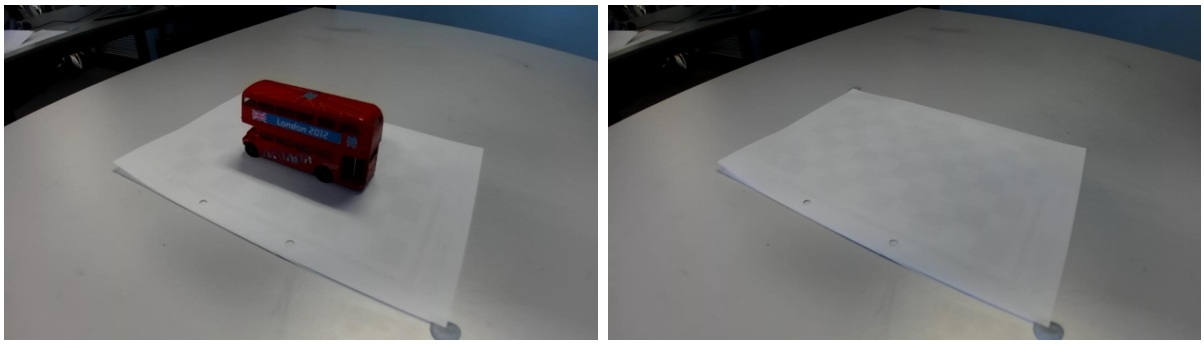
(a) Image of target object *(b) Isolated target object*

Figure 13: Example of blue-screen foreground isolation, showing effect of colour spill and shadows.

when performing the dataset augmentation process. There is a number of different ways in which foreground isolation can be performed. Initially, we attempted to use the chroma key method, also known as a 'green-screen' method. The method involves using a plain monochromatic background, typically either blue or green. The target object can then be isolated from the background by thresholding chroma values in the image. Green screens are typically more susceptible to 'colour spill', in which light reflects off the green screen and onto the target object, giving it a green tint. Due to their lower brightness and reflectivity, blue screens are less susceptible to this effect and because of this, we used a blue background to minimise unwanted colour spill.

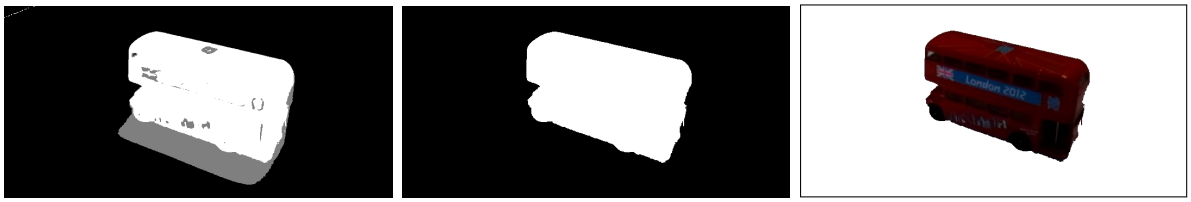
An example of a target object on the blue screen is shown in Figure 13(a), which reveals two issues. Firstly, colour spill is still evident in the cup, which appears to have a blue hue towards the base, despite being green and white. Also, the shadows near the base of the cup are very dark as the object is sitting directly on the blue table and is being lit from above. The chroma threshold must be set high enough to cover the variation in illumination of the blue screen. However, with this threshold, parts of the target object will be marked as background, which can be observed in Figure 13(b). Also, because the object border is not clearly defined in the image, we observe a blue fringe around the masked target object. Ultimately, due to the poor lighting in the workspace and the reflectivity of the blue screen on the target object, the chroma keying method does not produce results which are suitable to be used.

We attempt a different approach to foreground isolation - frame differencing. Noting that we can precisely and repeatably position the camera in the scene, we capture a set of library images with no object present in the scene. We then place the target object in the scene and capture the same set of images with the camera in the exact same set of poses. For a given pose, this provides us with two images, one of the target object, and one just



(a) Target object (foreground) *(b) Background*

Figure 14: Foreground and background images captured from a given pose.

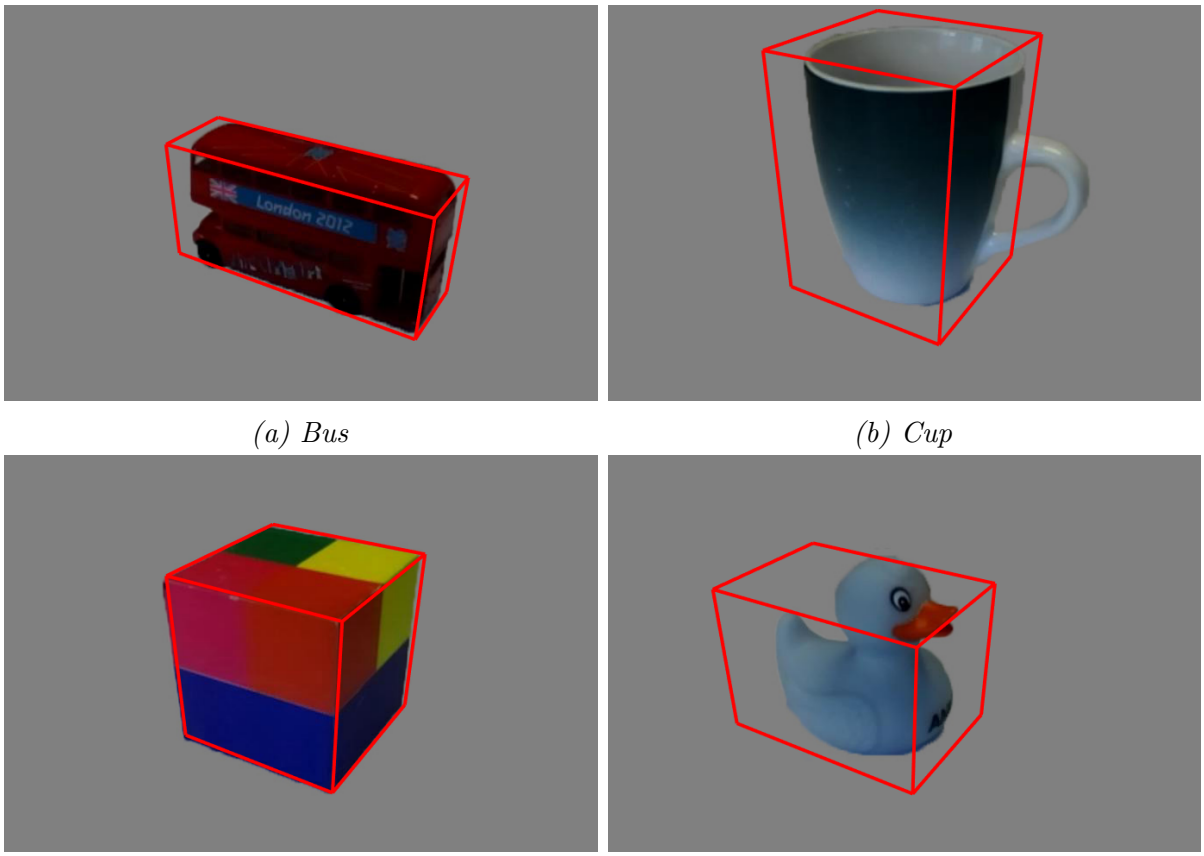


(a) Output from OpenCV *(b) Refined object mask* *(c) Extracted target object*

Figure 15: Intermediate steps of foreground isolation method using mixture-of-gaussian method.

containing the background. We present an example of these images in Figures 14(a) and 14(b). By calculating the difference between these two images, we can extract the target object from the background.

The performance of frame differencing is highly dependent on the function used to calculate the difference between the two images. Initially, we attempt a rudimentary difference function based on the Hue, Saturation and Luminance channels of the image and again, we encountered issues with shadows close to the base of the object. In search of a more effective algorithm, we choose the mixture-of-gaussians approach presented by Zivkovic [20] and implemented in OpenCV. We find the performance of this method is suitable for our use, especially with regard to detecting the difference between object shadows and the object itself, observable in Figure 15(a). We further refine the object mask produced by this method by using connected component analysis and morphologicial operation to identify the target object mask, fill in holes, and smooth the mask border. The results of this are shown in Figure 15(b).



(a) Bus *(b) Cup*

(c) Block *(d) Duck*

Figure 16: Example of data captured for object pose library. Ground truth is super-imposed.

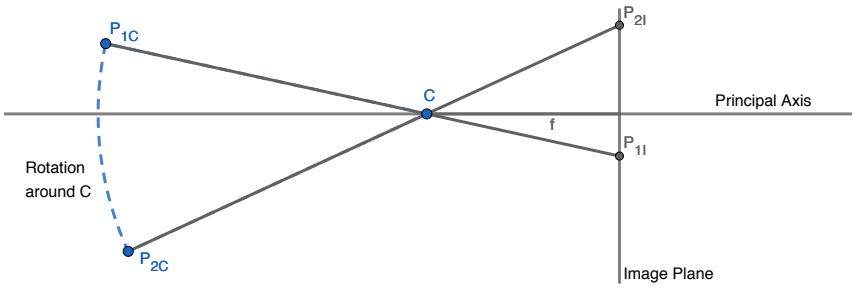


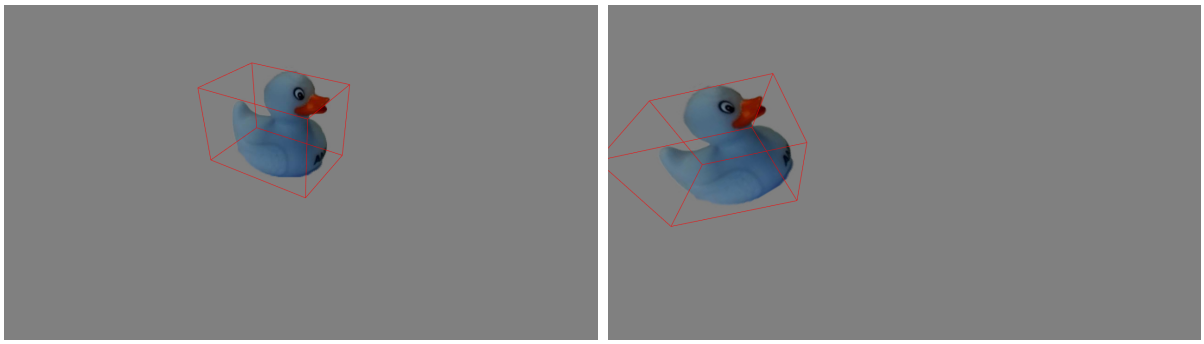
Figure 17: Diagram showing a point, P_1, being rotated about the camera principal point, C, resulting in P_2. The corresponding projections into the image plane are also shown.

5.6 Generating Training Images

The images that are captured from the experimental setup above only describe a small subset of poses that the target object could be in, and only offer a single view of each pose. In order to create a valuable training set for the purposes of training neural network pose estimators, many more images of the target object are required with much more variety in pose and appearance. However, capturing each image manually, or by using the robotic arm would be extremely time-consuming and would still not capture a wide enough variety of images. Thus, we seek to artificially augment the dataset using the data that has already been captured and add additional images for new poses or modified appearance of the object.

5.6.1 Simulating Object Translation

Images captured using the robotic arm are from poses which are distributed over a hemisphere centred on the target object. These poses are a fixed distance from the origin of the target object's coordinate system and the principal axis of the camera is pointed towards to origin of the target object. Thus, in all the captured images, the target object is in the centre of the image at a fixed relative size. We could significantly increase the size of the training set if we could virtually translate and rotate the camera into new poses. If we just consider a rotation of the camera about its principal point, we note that the camera's view of the scene does not change, only the projection of the scene onto the image plane changes. Thus, if we have an image of the target object, we can produce a new image from a rotated camera pose using only the original image. However, the rotation will result in a different projection into the image plane, which we can determine using the pinhole camera model. A diagram of this is shown in Figure 17.



(a) Original Image *(b) Simulated rotation*

Figure 18: Example of simulated rotation camera about the principal point.

The projection of a point, P_1, from the camera frame, C into the image frame I is represented as:

$$P_{1I} = \mathbf{K} \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} P_{1C} \tag{8}$$

If we define P_2 to be the rotation of P_1 about the camera frame origin, then

$$P_{2C} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} P_{1C} \tag{9}$$

$$P_{2I} = \mathbf{K} \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} P_{1C} \tag{10}$$

$$= \mathbf{KR} \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} P_{1C} \tag{11}$$

$$= \mathbf{KRK}^{-1} P_{1I} \tag{12}$$

This gives us a projective mapping, or homography, which we can use to warp the image to simulate a rotation of the camera. We can also apply the rotation to the ground truth of the object pose to determine the new ground truth for the image. An example of applying this technique is shown in Figure 18.

6 Discussion and Analysis

6.1 Background Replacement

Considering that the target objects have been isolated from the background, it makes sense to consider replacing the background with different images. In fact, this is a fairly common technique in machine learning to increase the size of the training set [4], [10]. It helps to ensure that the neural network does not use any properties of a particular background or scene to aid in determining the object pose. The standard way in which this is done is simply to isolate the target object, and superimpose it onto a random, generic image. While this does achieve the intended result of removing any context from the image, it results in extremely unrealistic training images. We consider the fact that the context surrounding a target object can be highly valuable when determining the pose of the object, especially its distance from the camera. If we consider the process a human takes when estimating the pose of an unknown object, it's common to take cues from the surrounding environment and nearby objects to indicate the relative size of the object, how the object casts shadows, and also and shadows that are cast onto the object. By simply superimposing the object onto a generic background, this information is lost, which can potentially have a detrimental effect on the effectiveness of training the neural network.

Instead of superimposing the target object onto random images, it would be desirable to insert the object into a different scene more realistically. For example, we could insert the target object into a different scene containing a table, and position the object such that it appeared to be sitting on the table. However, doing so without synthetically rendering a 3D model of the object into the scene is non-trivial. If the camera-to-world poses in each scene were identical this would be possible, but would drastically limit the number of usable scenes.

The most sensible way to achieve this kind of scene replacement would be to reconstruct a 3D model of the target object and then render it in the desired pose in any scene. This should be feasible given the object pose library that we have captured, however, as we have observed in many of the datasets in the literature, creating such a model generally requires human intervention if the model is to be accurate. This goes against the ethos of this project, which was to create this dataset with minimal human intervention and oversight. However, if we consider that this process would only have to be completed for objects in the training set, it may be worth considering. Under the proposed network structure, there would still be no requirement to create a 3D model for each target object

as only the object pose library is needed.

6.2 Simulating Occlusion

Another valuable inclusion to the dataset would be some occluded images of the target object. Again, there is a simple way and a more realistic way. The easiest way to generate occluded images would be to superimpose other objects onto a scene containing the target object such that the object partially occludes the target object. However, an issue similar to that in background substitution discussion is raised — how much context does the occluding object provide about the pose of the target object? One could argue that an occlusion situation provides information about the relative distances of the objects and that shadows could also provide additional context. If the occluding object is known, then it provides some additional information about the position and size of the target object. Thus, the argument could be made that the occluding object should be as realistically placed as possible. And, again, this leads to the same issue as discussed above. In order to be able to render the object in arbitrary poses, a 3D model is required — otherwise the number of poses that can be rendered realistically is severely restricted.

On the contrary, an argument can be made that a neural network should be trained to handle any arbitrary occlusion by objects which provide no context to the scene. This would in theory make the network more robust to different types of occlusion. It also provides a justification for simply rendering random objects on top of the target object without any regard for the geometry of the scene. However, the counter argument is that it is unproductive to train a neural network on examples that it is unlikely to encounter in the real world.

7 Current Limitations and Future Work

The immediate next steps for this project can be divided into two parts: the experimental setup and the data augmentation process. Firstly there are a number of different elements of the experimental apparatus which could be improved. The lighting conditions in the workspace were generally poor, as the room was only lit from the fluorescent ceiling lights and the ambient light from the windows. This lead to a number of unwanted shadows and generally poor lighting of the target object. This made the process of foreground isolation particularly difficult, especially with the very dark shadows close to the base of the object. A brighter, more even lighting setup would improve results significantly and make the process of foreground isolation much simpler.

The camera that was used to capture images was also an issue. In our testing, we were limited to a resolution of 1280×720 px, when it is actually capable of more than double this resolution. While it is likely that images would be down-sampled anyway when used as input to a neural network, a higher resolution would result in more accurate masks for foreground isolation, and more realistic images when warping and augmenting the images. The camera used also had a fixed focus and meant that if the camera was too close to the target object, then the images would be out of focus and blurry. Combined with the constraints of the robot arm, we were operating close to the limit of the focus range, which meant images were not as sharp as they could be. Also, considering that the camera is a stereo camera, it would have been possible to capture a depth map of the scene as well, however for simplicity and technical constraints we opted for only the RGB image.

In terms of data augmentation, there is no real limit to the amount of processing that can be done. However, as discussed above, the main limiting factor to this is the lack of an accurate 3D model of the target object. If there is an efficient and accurate way to reconstruct such a model using the data that we have captured, then this would enable a larger range of data augmentation than currently possible. It may be sufficient to simply capture a more densely-sampled set of pose library images, and render images with the closest matching image from the library.

Beyond just creating the dataset, there is a large amount of future work in implementing, training and testing a neural network that is structured in the way we have suggested. Whether such a network is feasible and can outperform the current state-of-the-art remains to be seen.

8 Conclusion

We explore the current state-of-the-art methods for 6D pose estimation. Based on our analysis of the literature, we identify the main deficiency in current machine learning techniques is that they involve a significant effort, both human and computational, to estimate the pose of new, unseen target objects. Based on this, we propose a new type of neural network structure which alleviates this issue by including an object pose library as a parallel input to the network. The main body of work focused on generating a new type of dataset to suit this network design, dividing our efforts into 3 separate phases: calibration, object pose library capture, and dataset augmentation. A 6DOF robot arm is utilised to provide accurate and repeatable positioning of the camera relative to the target object. We develop a MATLAB package to automatically perform both the camera calibration and hand-eye calibration of the robot. We then capture object pose library datasets for four different target objects and determine their corresponding ground truths. Using these images, we are able to isolate the target object from the background using a frame-differencing approach. We then explore a number of dataset augmentation techniques and provide a method to artificially rotate the camera view using existing data from the pose library. We discuss the viability of simulating both object occlusions and background replacement and conclude that, in order to do both of these effectively and realistically, a full 3D reconstruction of the target object would be required. The work presented here opens up a number of pathways for future work, and provides the foundational steps for a new type of network structure for performing 6D pose estimation.

References

[1] Boston Dynamics, *Atlas, the next generation*, Online, Feb. 2016. [Online]. Available: https://www.youtube.com/watch?v=rVlhMGQgDkY.

[2] E. Ackerman, *Aussies win amazon robotics challenge*, 2017. [Online]. Available: https://spectrum.ieee.org/automaton/robotics/industrial-robots/aussies-win-amazon-robotics-challenge.

[3] D. Morrison, A. Tow, M. McTaggart, R. Smith, *et al.*, "Cartman: The low-cost cartesian manipulator that won the amazon robotics challenge," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2018. DOI: 10.1109/icra.2018.8463191.

[4] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," Mar. 31, 2017. arXiv: http://arxiv.org/abs/1703.10896v2 [cs.CV].

[5] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, and D. Kraft, "BOP: Benchmark for 6D object pose estimation," *European Conference on Computer Vision*, 2018. [Online]. Available: https://arxiv.org/pdf/1808.08319.pdf.

[6] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, 2005. DOI: 10.1109/cvpr.2005.74.

[7] A. Sagitov, K. Shabalina, L. Sabirova, H. Li, and E. Magid, "ARTag, AprilTag and CALTag fiducial marker systems: Comparison in a presence of partial marker occlusion and rotation," in *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, SCITEPRESS - Science and Technology Publications, 2017. DOI: 10.5220/0006478901820191.

[8] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *2011 International Conference on Computer Vision*, IEEE, Nov. 2011. DOI: 10.1109/iccv.2011.6126326.

[9] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Computer Vision – ACCV 2012*, Springer Berlin Heidelberg, 2013, pp. 548–562. DOI: 10.1007/978-3-642-37331-2_42. [Online]. Available: http://www.stefan-hinterstoisser.com/papers/hinterstoisser2012accv.pdf.

[10] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6d: Making RGB-based 3d detection and 6d pose estimation great again," in *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Oct. 2017. DOI: 10.1109/iccv.2017.169.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd - single shot multibox detector," Dec. 8, 2015. DOI: 10.1007/978-3-319-46448-0_2. arXiv: http://arxiv.org/abs/1512.02325v5 [cs.CV].

[12] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," Feb. 23, 2016. arXiv: http://arxiv.org/abs/1602.07261v2 [cs.CV].

[13] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," *CVPR*, 2018. arXiv: http://arxiv.org/abs/1711.08848v4 [cs.CV].

[14] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6D object pose estimation," in *European Conference on Computer Vision*, Springer, 2016, pp. 606–619. DOI: 10.1007/978-3-319-49409-8_52. [Online]. Available: http://cmp.felk.cvut.cz/~hodanto2/data/hodan2016evaluation.pdf.

[15] T. Hodan, P. Haluza, S. Obdrzalek, J. Matas, M. Lourakis, and X. Zabulis, "T-less: An rgb-d dataset for 6d pose estimation of texture-less objects," *IEEE Winter Conference on Applications of Computer Vision*, 2017. DOI: 10.1109/WACV.2017.103. [Online]. Available: http://cmp.felk.cvut.cz/~hodanto2/data/hodan2017tless.pdf.

[16] C. Rennie, R. Shome, K. E. Bekris, and A. F. D. Souza, "A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place," *Unpublished*, Sep. 3, 2015. arXiv: http://arxiv.org/abs/1509.01277v2 [cs.CV]. [Online]. Available: https://arxiv.org/pdf/1509.01277.pdf.

[17] R. Tsai and R. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Transactions on Robotics and Automation*, 1989.

[18] Z. Lazarevic, *Hand-eye calibration tool - MATLAB*. [Online]. Available: http://lazax.com/www.cs.columbia.edu/~laza/html/Stewart/matlab/handEye.m.

[19] M. Deserno, "How to generate equidistributed points on the surface of a sphere," Sep. 2004.

[20] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition. ICPR*, IEEE, 2004. DOI: 10.1109/icpr.2004.1333992.