State Estimation for Robotic Harvesting Applications

Masters Thesis

Gerard Kennedy
U5185867

Student: Gerard Kennedy

Chair Supervisor: Prof. Robert Mahony

Associate supervisors: Dr. Xin Yu, Assoc. Prof. Nick Barnes, Prof. Hongdong Li

Australian National University

Acknowledgements

Firstly, thank you to my supervisor, Prof. Rob Mahony. His patience, encouragement and support have made this project both an enjoyable and rewarding experience. I am also grateful for the opportunity to be a part of his research group, and to work on a wide range of interesting projects. Thank you also to my associate supervisors, Dr. Xin Yu, and Assoc. Prof. Nick Barnes, and Prof. Hongdong Li for indulging my many questions and providing such sound guidance.

Thank you also to my fellow students, particularly Zheyu Zhuang and Pieter Van Goor, for their constant help, assurance, and advice. Being part of such a supportive environment was a highlight of my time at ANU.

Thank you to Dr. Viorela Ila, Dr. Tao Hu, and Alex Martin, for giving me the opportunity to be a part of our robotic harvesting project. I have benefited greatly from the opportunity of collaborating with and learning from you.

Finally, thank you to my partner Cass, for putting up with, and supporting me on this latest endeavour. It wouldn't have been possible without your support.

Abstract

The global population is expected to pass 9 billion by 2050, requiring ongoing improvements in food production methods. Robotic harvesting offers part of the solution to this challenge, and research into the use of agricultural robots (AgBots) for harvesting horticultural crops has increased over the past several decades. While there has been significant progress in harvesting automation for many crops, robotic systems for crops that require selective harvesting remain far from mature. A key roadblock for successful implementation of such systems is the need for high-speed, high-accuracy perception pipelines.

This thesis presents three projects which aim to develop techniques to enable implementation of fast, accurate robotic harvesting perception pipelines. These projects utilise state estimation techniques to solve common robotic harvesting perception problems.

The initial problem involves detecting, localising, and tracking green asparagus spears with the aim of selectively harvesting this crop. For this task the state is the location and height of each asparagus spear. As part of this work I also present a design for a prototype robotic platform for harvesting green asparagus.

The second application presented in this thesis is object pose estimation and refinement from a single RGB image. Object pose estimation is a key technique in many robotic harvesting perception systems, where it is used to determine the pose of the target crop. For this application a deep learning framework, titled the Innovation CNN, is developed. The Innovation CNN receives an image and an initial state estimate, and computes the gradient from the estimated state to the true state. The result is applied iteratively in a Stochastic Gradient Descent framework to refine initial pose estimates.

The third application presented in this thesis is depth estimation and refinement from a single RGB image. Depth estimation is regularly used in robotic harvesting applications as a means of determining crop distance/height, or as an intermediate step in computing a 3D reconstruction. The Innovation CNN is also applied to this problem, again using Stochastic Gradient Descent to refine the initial depth estimate.

Contents

List of Figures

List of Tables

Publications to Date

Conference (Accepted)

G. Kennedy, V. Ila, R. Mahony, *A Perception Pipeline for Robotic Harvesting of Green Asparagus*, 6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019

Submitted

G. Kennedy, Z. Zhuang, X. Yu, R. Mahony, *Iterative Optimisation with an Innovation CNN for Pose Refinement*, arXiv, available at: https://arxiv.org/abs/2101.08895

G. Kennedy, J. Gao, Z. Zhuang, X. Yu, R. Mahony, *Learning Innovations for State Estimation*, under review.

Chapter 1

Introduction

The United Nations has predicted that the global population will increase to over 9 billion by 2050, and that food production will increase by 70% in response [4]. It has also been observed that current agricultural practices have limited scalability due to factors including the lack of arable land, water shortages, under-investment, environmental considerations, and lack of available labour caused in part by an aging global population [5, 6]. The result is that we are faced with the challenge of producing more with less resources.

Agricultural robots (AgBots) have been identified as part of the solution to this impending global crisis [5]. AgBots are expected to fit into a broader picture in which crop breeding, planting, and harvesting practices are refined to allow greater levels of automation. Research into AgBots over the past three decades has led to the identification of agricultural applications that show promise for robotic automation, including vehicle guidance, crop monitoring, weed management, sowing, spraying, selective harvesting, and crop sorting [5, 7]. Increased levels of robotics in agriculture is predicted to improve efficiency throughout the food production chain.

While mechanisation of broad field farming is highly developed and robotic automation of these processes is improving, progress in selective harvesting (only harvest a crop if criteria are met) robotic systems has been slow [8]. There are still relatively few examples of successful prototypes and just one commercial example of a selective harvesting robot [8, 9]. A key reason for this slow progress is that most applications of selective harvesting AgBots require advanced sensor data processing to identify target crops for harvest amongst the unripe crops and surrounding plant matter [8]. Achieving reliable identification of crops remains a challenge in difficult on-farm conditions where speed and accuracy are paramount and the environment is highly variable [5]. For robotic technology to be successfully integrated into existing farm operations and logistics it needs to be able to operate at speeds and efficiencies comparable or superior to human labour. Current selective robotic

harvesters typically have cycle times in the range of 30 seconds per crop, significantly slower than human labour [8]. High speed harvest requires real-time (in the order of tens of milliseconds) perception and cognition as well as high speed actuators and robust mechanical designs. While actuation speed is closely tied to the choice of robotic manipulator and cutting mechanism, perception speed relies on a number of factors. These factors include data input device (depth/monocular/stereo camera(s), laser scanners, etc), choice of perception algorithms, and implementation of these algorithms. Accurate, high-speed perception is a key road-block to successful implementation and adoption of robotic harvesting platforms for selective harvesting. The aim of the perception system is to detect, localise, and track the target crop.

This thesis presents three pieces of work that apply state estimation and refinement concepts to robotic harvesting perception problems. In this context a 'state' is an internal representation of a system that is sufficient to fully define the future evolution of all system variables. The projects undertaken in this thesis aim to develop techniques for implementing fast, accurate robotic perception pipelines.

The first work involves applying classical algorithms from the fields of robotics and computer vision to the problem of detecting and localising green asparagus spears. In this work the state is a 3D object that includes the 2D position and 1D height of asparagus spears. This is in contrast to typical approaches to robotic harvesting, which often model the state as a 6D pose (3D position and 3D orientation). This reduction in dimensionality is employed to simplify the vision algorithm, for which speed is a crucial design criteria. The reduction is possible due to the simplicity of the problem when compared with other types of crops (relatively little surrounding plant matter). The estimated state is tracked and updated across an image sequence using a Kalman filter as the harvester moves over the target crop. The prototype state estimation pipeline developed for this problem is presented in Chapter 4.

The state estimation pipeline was developed as part of a larger project encompassing the development of a robotic harvesting platform targeting green asparagus. The platform is designed to optimise speed, and to allow for easy integration into existing farm systems. This platform is also presented in this chapter. The market opportunity for a robotic harvester of green asparagus is also discussed in this chapter due to the initial aim of developing a platform for commercial use. For a number of reasons including funding complications and personnel changes the goal of developing a prototype robotic harvesting platform was deemed infeasible during the course of this project. This is discussed further in this chapter. This outcome led to a change of research direction toward the more general problem of state estimation

and refinement, with the initial target problem of object pose refinement.

The second work presented in this thesis develops an iterative framework for object pose refinement. Object pose estimation is a crucial task for robotic harvesting applications, as an inaccurate pose can lead to crop/environment damage and/or reduced harvesting success rates. A general state refinement framework for object pose refinement is applicable to any crop type, regardless of specific characteristics. In this thesis the framework is applied to generic objects from common pose estimation datasets, including toy animals such as apes and ducks. Such objects are similar to crops in several ways. Crops and these objects typically have at least one plane of symmetry and are typically a certain colour, while not having clear distinguishing features such as a handle. Such objects are challenging for robotic manipulation systems, due to their variability and required dexterity when they are manipulated. The task is particularly challenging for crops, where the object is easily deformed and damaged. The requirement for a high level of accuracy for robotic harvesting applications motivated the development of this framework.

The framework involves a novel Convolutional Neural Network (CNN) that is trained to estimate the gradient between a provided state and the true state. The algorithm is named the Innovation CNN, as 'innovation' is a term used in control theory, where it refers to the difference between the predicted and desired measurements. In this work an initial state estimate is obtained form a separate, baseline network trained for the task of object pose estimation. The output of the baseline network is the target state for this application. The Innovation CNN then estimates the gradient between the given state and the true state. The innovation is then applied iteratively in a Stochastic Gradient Descent (SGD) framework to refine the initial estimate. This work is the focus of Chapter 5.

The third work presented in this thesis applies the state refinement framework discussed above to the problem of depth refinement. This work was undertaken to demonstrate the general applicability of the Innovation CNN to state refinement problems. An accurate depth map is often crucial to robotic harvesting, as depth can be used to determine distance to, or height of a target crop, and also as in intermediate step in a 3D scene reconstruction. For this problem a similar approach is used, in which an initial state estimate is refined iteratively using the Innovation CNN. In this case the state is defined by the 1D per-pixel depth map. This work is the focus of Chapter 6.

Of the three projects presented in this thesis, the first is developed specifically for a robotic harvesting platform. The following two projects are developed as general approaches to common robotic harvesting perception problems. The remainder of

Figure 1.1: Typical asparagus bed [1].

this introduction is split into two sections according to this distinction. The asparagus harvester and the corresponding state estimation pipeline are introduced in Section 1.1, while the state refinement framework is introduced in Section 1.2.

1.1 A Robotic Harvester for Green Asparagus

Green asparagus is an attractive target crop for a robotic harvesting platform due to the relatively low levels of surrounding foliage, which leads to less occlusion to the vision system and easier harvester control. Spears grow straight out of the ground with no obscuring foliage (see Figure 1.1). The crop grows irregularly in beds with the harvest period covering eight to ten weeks in Australia. Asparagus spears are harvested when they reach between 20 to 25 cm tall. As asparagus spears can grow up to 5cm per day, each row must be harvested each day during the peak growing period, and only those asparagus in the correct height range should be harvested.

The state estimation algorithm presented in this thesis for tracking green asparagus was developed within the context of a wider project aimed at robotically harvesting this crop. The robotic harvesting platform developed for this task has three main systems: manipulation, transportation, and perception.

The manipulation system involves a novel cut-and-throw approach to harvesting implemented with a Delta Robot. The Delta Robot can operate at speeds up to 10m/s and accelerate at 10m/s^2. The combination of speed with the simplicity of

a single robotic manipulator allows for harvesting speeds of approximately 3 spears per second, which significantly exceeds current state of the art performance.

The transportation system involves a tractor-mountable enclosure. This allows for easy integration of the platform into current farm operations, while reducing up-front costs. The target is for the robotic harvest system to travel at 1-2 m/s, roughly an order of magnitude faster than existing systems and comparable or superior to human labour.

The physical perception system consists of 3 Point Grey monocular cameras. The software that uses incoming image information to track asparagus spears is referred to as the 'perception pipeline' or the 'state estimation pipeline' in this thesis. The pipeline estimates and tracks a 'state' containing the position and height of visible green asparagus spears. The pipeline is optimised for speed and accuracy through the use of multiple cameras, a distributed compute architecture, and simple algorithms. In this thesis results are presented that demonstrate the performance of the pipeline on various crop layouts in-lab, and in difficult on-farm conditions.

The key contributions of this thesis related to asparagus harvesting are:

- a prototype robotic platform designed to harvest green asparagus spears at high speeds
- the market opportunity for a green asparagus harvesting robot in Australia
- an algorithm architecture capable of calibrating a multi-camera system, and detecting and localising asparagus spears in outdoor conditions
- a novel single-image representation of a scene captured by a multi-camera system
- on-farm and in-lab trials of the perception pipeline

1.2 A Deep Learning Algorithm for State Refinement

Motivated by the requirement of robotic harvesting systems to have an accurate estimate of the position and orientation of target crops, this thesis presents a general framework for object pose estimation and refinement. Object pose is a 6-dimensional object containing 3D position and 3D rotation of target objects relative to a chosen reference frame. This problem is of particular important in robotic harvesting applications, as an accurate estimate increases the likelihood of harvest success and reduces the likelihood of crop damage. Object pose estimation in a harvesting context also presents different challenges to those typically presented in computer vision

datasets. For example, when dealing with crops each instance of the object is different. Each banana is only an image of the ideal form of a banana. This differs from standard datasets, which show the same object under different levels of occlusion and in different poses. Crop objects also typically have planes of symmetry, which increases the difficulty of obtaining an accurate pose estimate. For these reasons, this work focuses on refining initial pose estimates to obtain more accurate results, and particular attention is paid to standard objects that contain symmetries and have low surface texture.

Object pose estimation and refinement is an important problem in computer vision, and has a range of applications beyond robotic harvesting, including general robotic manipulation/grasping, and virtual/augmented reality. Recent state-of-the-art object pose estimation algorithms use a two phase approach, initially providing a rough estimate of pose, and then using a second network to refine the pose estimate in order to obtain the desired performance [10, 11]. The principle of refining an existing estimate of pose can be applied multiple times [12, 11, 13]. Such an approach can be seen as analogous to iterative state refinement algorithms such as SGD that are well established in the optimisation field [14].

In this thesis established principles of SGD are drawn from to formulate a novel CNN architecture to implement an iterative refinement based on formal principles. In doing so, the requirement for the neural network to accurately estimate the pose in a single forward pass is eased. Specifically, the loss function for a recent pose estimation network is modified to estimate an update or refinement to an existing state, rather than estimating the state directly. An autoencoder network is implemented in parallel with the modified pose estimation network architecture to encode the state estimate and inject this information into the modified pose estimation network using skip connections. This overcomes the requirement to reconstruct an image from the state information in order for the state information to be injected. The output of the modified pose estimation network is used as an SGD update for the state estimate. By formally recognising the role of the state estimate and the SGD update term, existing algorithms and insights on step-size choice and convergence analysis that have been developed for SGD can be employed.

This approach is applied using a recent network for object pose estimation, PVNet [15]. The target state is the output of this network, which is a vector field representation from which object pose can be optained via a perspective-n-point (PnP) algorithm. To measure the performance of this approach two widely used benchmarks for 6D object pose estimation, LINEMOD [16] and Occlusion LINEMOD [17] are used. The proposed approach refines initial pose estimates provided by PVNet,

(a) Initial Estimate (b) Iteration 1 (c) Iteration 2 (d) Final Estimate

Figure 1.2: The proposed method applied to the Ape object of the **LINEMOD** dataset. Green bounding boxes represent ground truth poses and blue boxes represent the result of the proposed method.

leading to an average improvement of 4.20% on LINEMOD and 8.12% on Occlusion LINEMOD relative to the baseline network, in terms of the widely-used ADD(-S) metric. This results in state-of-the-art results performance on both datasets. Results also indicate that this approach is particularly effective at refining relatively poor initial estimates. Specifically, relatively large improvement can be seen on the more difficult Occlusion LINEMOD dataset, along with particularly challenging objects of the LINEMOD dataset, such as ape (21.14% improvement) and duck (10.73% improvement) where the objects have poor surface texture. An example result illustrating the iterative improvement achieved by the proposed approach is shown in Figure 1.2.

The Innovation CNN is a state estimation technique that shows promise for a range of robotic applications. To demonstrate the general utility of the Innovation CNN as a method for learning and iteratively refining an initial state, the same approach is also applied to the problem of depth estimation. Depth is an attractive problem for this technique, as CNNs for depth estimation generally regress directly to per-pixel depth values. This is in contrast to object pose estimation, for which the CNN regresses to an intermediate representation from which object pose is obtained via a secondary process. The Innovation CNN also inherently contains discontinuous detail at the pixel level, which occurs at occlusions and object edges. These details are difficult to resolve in a single forward pass of a CNN and can therefore be significantly improved via iterative refinement with the proposed approach. Depth estimation is also regularly required in robotic harvesting perception pipelines, where it is used to measure crop height or distance, and as an intermediate step in a 3D reconstruction pipeline.

Depth refinement is performed with a similar approach to the previous formulation. The loss function for a recent depth estimation network, FastDepth [18], is

modified, and the network output is applied in an iterative SGD framework to refine an initial depth estimate. This approach is evaluated using the NYU Depth V2 dataset [19], with which a relative improvement of 4.58% and 4.86% is obtained in terms of the standard metrics.

In summary, the key contributions of this thesis related to iterative state refinement are:

- a novel, general framework that uses a CNN to estimate an innovation term and uses this term to refine an initial state estimate via SGD
- a demonstration that this framework improves the results of an off-the-shelf object pose estimation network
- a demonstration that this framework improves the results of an off-the-shelf depth estimation network

1.3 Thesis Structure

Following this introduction, the remainder of this thesis is structured as follows: Chapter 2 presents a review of the relevant literature for robotic harvesting and state estimation. This includes classical approaches to state estimation and refinement, and state estimation as a supervised learning problem. Particular emphasis is placed on the target problems of object pose estimation and refinement and depth estimation and refinement. Chapter 3 discusses the commercial opportunity for a robotic platform for harvesting green asparagus, and provides an overview of the proposed platform. Chapter 4 focuses on the state estimation pipeline developed for the green asparagus harvesting robot. Chapter 5 presents the approach and results of learning a state gradient for an object pose refinement problem, and applying this output in a gradient descent framework to improve initial pose estimates. Similarly, Chapter 6 presents the approach and results of learning a state gradient for depth refinement and applying the gradient in an iterative framework. Finally, Chapter 7 discusses possible next steps.

Chapter 2

Literature Review

The following review is arranged into five sections. In Section 2.1 key concepts including 'state', 'state estimation', and 'innovation' are introduced. Literature related to the general problems of online and offline state estimation is also outlined. In Section 2.2 literature related to robotic harvesting is discussed, with particular emphasis placed on robotic harvesting approaches that incorporate computer vision techniques. The commercialisation pathways for key recent agricultural robots (AgBots) are also outlined. In Section 2.3 literature related to the problems of object pose estimation and refinement from a single RGB image is discussed. In Section 2.4 literature related to the problems of depth estimation and refinement from a single RGB image is discussed. Finally, Section 2.5 summarises recent approaches to state estimation that combine data-driven and knowledge-driven techniques, providing context for the approach taken in this thesis.

2.1 State Estimation

Central to all projects undertaken for this thesis is the concept of *state*. In control theory, a state is an internal representation of a system that is sufficient to fully define the future evolution of all system variables [20, 21]. A *state estimation algorithm* is an algorithm that enables the extraction of information about features of a system that are not explicitly provided by the available data [21]. The problem of obtaining a state estimate is relevant to applications within a variety of areas, from robotics and computer vision to economics and finance. For example, in robotics a state estimation algorithm could estimate the motion of the robot, given information from the robot's sensors. Likewise, many applications within computer vision involve estimating attributes of a scene, such as 3D structure, from a given image. Such attributes can also be considered a state. These two examples highlight a key distinction between state estimation applications: those applied online and those applied offline. Of-

fline estimation refers to problems involving a fixed input, typically an image in the relevant applications, from which information such as depth, camera or object pose estimation, or object detection can be estimated. Conversely, online state estimation, often also referred to as 'filtering' or 'observing' in robotic vision literature, involves estimating a state based on the previous state and current sensor input in an temporal process [22, 21]. The types of algorithms used for state estimation are generally different for each of these two application categories. An overview of applications of both online and offline state estimation is included below.

2.1.1 Offline State Estimation

Offline state estimation is at the core of many problems in computer vision. Such problems typically receive image data as input and estimate features of the scene such as object type or pose, depth, or 3D structure. As the information comes in the form of images a particular algorithm architecture, the Convolutional Neural Network (CNN) is widely used in such applications. CNNs attempt to 'learn' a function that best approximates the underlying data distribution present in the provided image. This is a challenging proposition, as state spaces of most learning problems are very large, non-linear, and high-dimensional. This is challenging because the algorithm must learn to converge from a wide range of initial conditions to a very precise result. For this reason it is often helpful to separately learn a state update that can be applied to the initial estimate as a refinement. In this way the search space is constrained by the training data to the region of the initial estimate. This approach has recently proved beneficial in research areas such as object detection and object pose estimation [11, 13, 12].

All applications involving static input information, such as an image, from which implicit information is extracted can be considered an application of offline state estimation. Such applications include object detection, depth estimation, deblurring, super-resolution, denoising, and object pose estimation, [23]. A common approach to such problems is to formulate them as either classification or regression problems, which can then be solved in a single step using a deep neural network (often a CNN). Such an approach is an example of data-driven modelling, as the algorithms used to estimate the state rely on training data for their accuracy. Classification and regression via a Deep Neural Network has become a very popular approach to offline state estimation in computer vision in the previous two decades. This revolution is largely due to the current abundance of training data, and the ever-increasing capacity of computational resources. An overview of some example offline state

estimation applications is provided below.

Object detection involves locating and classifying objects in an image and labeling them with a bounding box indicating detection confidence [24]. The two typical approaches to object detection involve either a) generating region proposals, extracting high-level features, and classifying the regions [25, 26], or b) directly via either regression [27, 28] or classification [29] with a single network. Approach a) is viewed as the traditional method, while method b) has been introduced mainly to allow for real-time applications.

Deblurring involves removing Gaussian blur and motion blur from an image [30]. Super-resolution involves restoring high-resolution images from one or more low resolution images [31]. Denoising involves restoring an image corrupted by Gaussian noise to its original form [32]. In each case, a widely used technique is to use a CNN to regress to the desired state in an end-to-end framework. Thus it appears feasible to reformulate these problems into the proposed iterative refinement framework.

Two more examples of offline state estimation are object pose estimation and depth estimation. As these are two target applications of this work, they are discussed in greater detail in Sections 2.3 and 6.1.1 respectively.

2.1.2 Online State Estimation

Online state estimation is fundamental in robotic vision, as robots generally have to function in a dynamic environment. Relevant problems involve continuously updating a state estimate as new information becomes available from the robot's sensors. Algorithms that have been developed for such problems are often referred to as 'filters' or 'observers' [21]. Such algorithms typically have an *innovation* term. In control theory, an innovation term measures the difference (error) between the estimated and true measurements, thus providing a quantification of the new information obtained from the latest measurements, in addition to the best existing state estimate. This term is used to correct the state estimate [33, 21]. In classical approaches the innovation is computed analytically rather than learnt, as the underlying state space is much more constrained. Applications in this category include visual odometry (VO), optical flow estimation, and simultaneous localisation and mapping (SLAM).

Visual odometry (VO) involves estimating the egomotion of an agent using only the input from one or more cameras, and is updated iteratively as new image data becomes available [34]. VO is generally solved by computing relative camera pose via 3D-to-2D correspondences and a PnP algorithm. Currently popular approaches

to VO include Sparse Visual Odometry (SVO) [35], which estimates relative camera pose by minimising photometric error across image patches, and Discrete Visual Odometry (DSO) [36], which minimises photometric error across image features and jointly optimises for camera intrinsics, extrinsics and inverse depth. There has also been recent interest in approaching VO from a learning perspective. Such work includes [37], which infers pose directly from a sequence of images using a Recurrent Neural Network (RNN). Similarly, [38] constructs a VO algorithm by combining CNNs for depth and velocity estimation.

Optical flow (OF) estimation involves estimating per-pixel motion between consecutive frames in an image sequence. OF has traditionally been approached as an optimisation problem based on hand-crafted image features [39]. Recent work [40, 41] have cast OF as a learning problem, using iterative residual refinement and gated recurrent units respectively. [40] is particularly similar to the proposed approach, as they learn the residual and then reapply the same network to iteratively improve the estimate. [40] is discussed further in Section 2.5.

The simultaneous localisation and mapping (SLAM) problem extends VO to include an estimation of the environment state. This requires a large state vector, of the order of the number of landmarks. Solutions to the SLAM problem typically involve a Bayesian filter such as the Extended Kalman Filter [42], or optimisation-based [43, 44]. Recently a new approach has emerged, formulating SLAM as a non-linear observer problem [45, 46]. The advantage of this is it removes the necessity to explicitly define a map origin. Like previous applications, there have also been recent efforts to apply learning to the SLAM problem. In some cases learning has been incorporated as front-end feature extraction [47, 48] for an optimisation-based back-end. Efforts have also been made to develop an entirely deep learning-based SLAM pipeline [49].

2.2 Robotic Harvesting

This section presents an overview of robotic harvesting literature, with a focus on platforms that target green asparagus.

The first example of a computer vision application to agriculture involved utilising light reflectivity differences to detect citrus fruit [50]. The first applied system was developed in 1977 for apple detection, using a black and white camera with red optical filter along with intensity thresholding and morphological operations [51]. Early applications of computer vision to crop detection utilised mainly spectral, intensity or range cameras, and pixel-wise or shape-based crop detection methods [52].

A survey of robotic harvesters conducted in 2013 of 50 distinct projects over the previous 30 years found that average harvesting success rate was 66%, and average cycle time between harvests was 33 seconds [53]. A review of harvesting robots in high value crops undertaken in 2013 found that localisation success across the reviewed robots was 85%, while harvesting success was 66% [54]. This indicates that 15% of the crop is missed before harvesting is even attempted.

A recent review of selective harvesting robots [8] found that perceiving the target crop remains a significant barrier to acceptable harvest success rates. Specifically, they discuss several examples of recent robotic platforms which have harvesting success rates of between 61-76.5% in an unmodified crop environment, and success rates of between 18-47% in unmodified conditions. This indicates that significant perception improvements and/or crop environment adaptions are required before acceptable harvesting success rates can be reached. Due to the sparse, unstructured nature of on-farm crop environments, the perception pipeline plays a critical role in the success or failure of any selective asparagus harvester [8].

Robotic harvesters are currently being developed for a range of crops that require selective harvesting, with oranges, tomatoes, apples, and asparagus being among the most frequently targeted [53]. A majority of these systems include a bespoke transportation system involving wheels [55, 56, 57] and often require additional infrastructure such as rail tracks of overhead wires [58]. However, some platforms make use of existing farm equipment, usually by developing a platform to be towed behind a tractor [59, 60, 9]. Common approaches to crop manipulation include using a suction device [56, 61] or gripper [62] to hold the object, and a second device to perform the harvest. Such systems therefore typically require two robotic manipulators, although there are exceptions such as [56, 59], for which both devices are situated on the one manipulator. The use of two interacting manipulators greatly increases the complexity of the harvesting task.

Perception systems for recent robotic harvesters have included single RGB [56], RGBD [63], and multi-camera + 3D sensor [64]. These systems are often combined with 3D reconstruction and 2D or 3D segmentation algorithms [63, 61]. A recent harvesting platform for green asparagus [63] generates a point cloud, which if followed by template matching to detect asparagus. In the past decade a number of solutions have incorporated deep learning techniques into the perception pipeline [62, 65, 61]. For example, Harvey the capsicum harvesting robot incorporates a CNN for 2D peduncle detection, combined with 3D reconstruction followed by pose estimation [65]. Similarly, [66] and [62] both implement CNN's to perform crop classification. Relative to many selectively harvested crops, green asparagus offers a simple perception

problem due to the limited surrounding foliage [63].

Previous green asparagus harvesting robots have typically employed either LIDAR and/or single camera sensors. These sensing modalities have led to perception systems that only provide information about asparagus height (LIDAR) or else rely on computationally expensive algorithms such as dense 3D reconstruction. The first green asparagus harvester, CAMIA, was developed in the 1990's using laser sensors to detect asparagus height. While CAMIA had promising initial results, unreliability of the sensors caused many asparagus to be missed and damage caused to the bed [67]. Similar to CAMIA, the Kim Haws harvester [68] and Geiger-Lund harvester [60] currently being developed also employ laser sensors to detect crop height, and have also been noted to miss a significant proportion of ripe asparagus [63]. MANTIS [69], utilises a LIDAR sensor suite and can detect asparagus at a rate of 1 every 6 seconds [69]. The GARotics harvester employs an RGBD camera and 3D reconstruction to harvest asparagus and reported a 90% harvest success rate of detected spears, although their detection success rate is not included [63]. Finally, recent work by The University of Waikato (UW) and Robotics Plus Limited has led to the development of a prototype harvester [70, 71, 72, 73, 59]. In this work a prototype robotic harvesting platform is presented that is shown to be capable of harvesting 92.3% of detected spears, while operating at 0.33m/s. While this is still slow relative to human labour, they note that their vision system appears capable of operating at 1m/s with more sophisticated software. The vision system for this platform incorporates time-of-flight and RGB cameras along with CNN algorithms to perform asparagus detection.

Figure 2.1 provides an overview of current/previous robotic harvesting platforms for green asparagus. Table 2.1 provides a summary of the technical approach of current/previous projects targeting robotic harvesting of green asparagus. Table 2.2 provides an overview of the state of development of current robotic projects targeting harvesting of green asparagus.

	Our Approach	Advantage	Other Asparagus Robots
Robotic Platform	6 d.o.f Delta Robot	fast, repeatable action, reduced harvesting time	PLC actuators [63], Servo controlled manipulation [69] 6 d.o.f robot arm [67]
Harvesting Action	cut/throw	reduced handling, faster harvesting	Pick and place [67, 63, 69, 59]
Vision System	multiple cameras	limited occlusion, high resolution data	1D laser scanner [67], 2D laser scanner [69], 2 laser scanners, 1 camera [74], 1 stereo camera [63] 1 TOF and 1 RGB camera [59]
Perception Approach	2D segmentation, homography projection, line fitting	high speed, continuous calibration	raw laser information [67], mean-filtered laser scan information [69], 3D reconstruction [74], 3D reconstruction and bundle adjustment [63], 3D reconstruction and CNN-based segmentation [59]

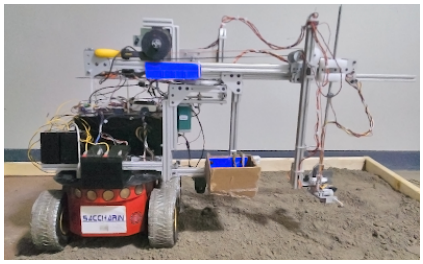
Table 2.1: Comparison of Robotic Green Asparagus Harvesting Approaches.

Robotic Harvester	Operating in Australia	Market Status	Country of Origin	Development Stage
GARotics [75]	No	Yet to reach market	Germany	TRL 7
MANTIS [69]	No	∼200 units sold	USA	Early Production
Kim Haws [68]	No	Yet to reach market	USA	∼100 prototypes available
Geiger-Lund [60]	No	Yet to reach market	USA	On field prototypes available
CAMIA [67]	No	Never made it to market	Australia	Prototype developed 2015
UW & Robotics Plus [59]	No	Yet to reach market	New Zealand	Proof-of-concept developed 2020

Table 2.2: Selective Harvesting Robots Developed for Green Asparagus.

(a) CAMIA Harvester [67].

(b) Kim Haws Harvester [68].



(c) Mantis Harvester [69].

(d) Geiger-Lund Harvester [60].

(e) GARotics Harvester [57].

(f) UW & Robotics Plus Harvester [59].

Figure 2.1: An overview of current/previous robotic harvesting platforms targeting green asparagus.

2.3 Pose Estimation

Object Pose Estimation from a single RGB image is an example of offline state estimation typically performed in a single step with a CNN. Obtaining an accurate pose of a target object within an image has a range of real world applications, including robotics, scene understanding, augmented reality, and virtual reality. This problem is made easier if RGBD images are utilised, however, lack of reliability and environmental constraints mean that this is not always possible. Pose estimation from a single RGB image is therefore an important research problem. It is also a highly challenging problem due ambiguity inherent in the visual appearance of objects from

different viewpoints. Due to symmetries objects often appear the same from a range of viewpoints. Research in object pose estimation can be sub-divided into four general areas since the deep learning revolution. These areas are: end-to-end regression from image to pose, pose classification via a discretised pose space, regression to an intermediate representation such as keypoints, and pose refinement. These areas are discussed in detail below.

2.3.1 End-to-End Regression

End-to-end pose regression from a single image is a highly challenging task due to the high dimensional, nonlinear space of rigid 3D rotations $SO(3)$. This is addressed by Kendall *et al.* [76] by separating and carefully balancing translation and rotation terms in the loss function, although this is for camera pose estimation, a slightly different problem. More recently, Xiang *el al.* [77] proposed to decouple or 'disentangle' translation and rotation terms by moving the centre of rotation to the centre of the object and representing translation in 2D image space. In practice, regression to 3D orientation has had limited success [78, 79]. This is partly due to such methods only covering the small subsection of pose space that is seen during training [10], which has in turn led to discretisation of the pose space as discussed below. Recently, [80] achieved state-of-the-art via simultaneous regression to object class, bounding box, rotation, and translation.

2.3.2 Pose Classification

Pose classification has typically involved discretising $SO(3)$, while the translation component is obtained via regression. However, the dimensionality of $SO(3)$ still provides difficulty, as even a coarse discretisation of $\sim 5^\circ$ leads to over 50,000 possible cases [79]. Kehl *el al.* [10] propose SSD6D, which decomposed 6D pose into viewpoint and discretised in-plane rotation. However, as noted in [79], this approach leads to ambiguous classifications as a change of viewpoint can be nearly equivalent to a change of in-plane rotation. Furthermore, such classification leads to coarse pose estimates that require further refinement [10, 12], as discussed below.

2.3.3 Pose via an Intermediate Representation

A recent, popular approach to pose estimation is to first regress to an intermediate representation such as keypoints, from which pose can be obtained via 2D-3D correspondences and a PnP algorithm [81, 82, 15, 11, 83, 84]. Of these approaches, some,

including Tekin *et al.* and Rad *et al.* [81, 82] regress to a set of bounding box corners. Such methods encounter difficulty when objects are occluded or truncated. Alternatively, Peng *et al.* [15] propose PVNET, which predicted pixel-wise unit-vectors that in turn indicate direction to keypoints. Similarly, Zakharov *et al.* [11] propose DPOD, which first applied a texture map to objects with a 2D image generated from spherical or cylindrical projections, and then used these to estimate dense correspondences. More recently, Song *et al.* [84] propose HybridPose, which implemented the vector directions from PVNET [15] while also estimating object edge vectors and symmetry correspondences.

2.3.4 Pose Refinement

Offline state estimation does not necessarily need to be performed in a single step. For some applications, it is helpful to iteratively step towards the optimal solution, thus reducing the search space at each iteration. Such problems typically involve utilisation of first order optimisation algorithms such as gradient descent, or second order optimisation algorithms such as Newton's method. An application in this category involves determining the robotic joint angles that provide a desired configuration. This problem is known as inverse kinematics, and is typically solved via either the image Jacobian, or a numerical optimisation approach such as Newton's method [85]. Beyond robotic vision, applications of numerical optimisation are numerous, ranging from investment portfolio management to computing the optimal shape of mechanical components [86]. Due to the widespread applicability of these methods, substantial research has been devoted to developing them [86, 87].

Recently, neural networks have been applied to problems typically solved with numerical optimisation, such as inverse kinematics [88], and 3D reconstruction [89]. Similarly, neural networks have been employed to refine a state estimate provided from a preceding algorithm, such as a different network. Applications that have made use of this approach include the initial target application, pose estimation, as discussed below.

Of the networks discussed above in Section 2.3, several have additional refinement steps that can be added to improve results. For example, Li *et al.* [12] presented DeepIM as a refinement step for PoseCNN [77]. Likewise Manhardt *et al.* [13] proposed a refinement step for SSD6D [10]. Finally DPOD [11] is presented along with a refinement step. All these methods perform refinement using the additional input of a 3D model of the target object. In each case a synthetic image of the target object is rendered using the initial pose estimate and the 3D model. The key difference in

these approaches is in the representation of the relative pose. DeepIM [12] utilises an 'untangled representation' in which the centre of the object is the centre of rotation, and translation is estimated in pixel space. Manhardt *et al.* [13] represent rotation as a unit quaternion and translation as a vector in $\mathbb{R}^3$. DPOD [11] combines these approaches by estimating rotation relative to the object centre, and estimating translation as a vector in $\mathbb{R}^3$. These approaches therefore differ mainly in their loss functions, and all require annotated 3D object models. Such networks are conceptually similar to the network proposed in this report, namely, they receive an input image and a pose estimate, and return an updated pose estimate. Unlike these approaches however, our method does not require an additional 3D object model, and can be applied more generally to a variety of applications not limited to pose estimation. It also iteratively improves the pose estimate based on established optimisation techniques rather than via visual similarity. It is therefore feasible to take the refined estimate produced by our network and refine it further using a pose refinement network which utilises the additional information of the 3D model.

2.3.5 Benchmarks and Datasets

Popular datasets for object pose estimation include LINEMOD [16], Occlusion LINEMOD [17], and YCB-Video [77]. Popular metrics for this task include the average distance (ADD) metric proposed by [16] (or ADD(-S) for symmetric objects), and the 2D projection metric which computes the mean distance between 2D projections of 3D model points. These metrics are presented in greater detail in Chapter 5. A pose is considered 'correct' in terms of the ADD metric if it is less than 10% of the models diameter, or in terms of the 2D projection error if it is less than 5 pixels.

2.4 Depth Estimation

Depth estimation from a single RGB image, often referred to as monocular depth estimation, is another example of offline state estimation typically performed in a single step with a CNN. Obtaining an accurate per-pixel depth map from a given image has a range of real world applications, including robotics, scene understanding, augmented reality, and virtual reality. The problem is challenging due to the inherent scale ambiguity. For obvious reasons this problem is trivial if RGBD images, LIDAR, or stereo cameras are utilised, however, lack of reliability and environmental constraints mean that this is not always possible. Monocular depth estimation is therefore an important research problem.

Research in monocular depth estimation can be sub-divided into three general areas since the deep learning revolution. These areas are: supervised estimation, semi-supervised estimation, and depth refinement. These areas are discussed in detail below.

2.4.1 Supervised Depth Estimation

Supervised depth estimation involves the use of per-pixel ground truth depth maps. Deep learning was first applied to the problem of monocular depth estimation by Eigen *et al.* [90]. In this work, two CNNs are used, the first provides a coarse depth estimate which is then locally refined by the second network. This work also introduced a scale-invariant mean-squared error to deal with scale ambiguity.

Liu *et al.* [91] combine the concepts of deep learning with continuous conditional random fields (CRFs) and propose a deep structured learning scheme which learns the unary and pairwise potentials of continuous CRFs in a deep CNN framework. They use a CNN to learn the relations of neighbouring superpixels, and learn the potentials of a CRF. Li *et al.* [92] proposed a deep learning architecture for coarse depth estimation which is then refined via a CRF. Eigen and Fergus [93] build upon the previous work of Eigen *et al.* [90] by proposing a single network that uses a sequence of three scales to generate and refine predictions to a higher resolution.

In recent years the problem of monocular depth estimation has received significantly more attention. Some key work includes Kendall *et al.* [94], who show that depth estimation and generalisation can be improved by utilising semantic labelling multi-task learning. Fu *et al.* [95] address the slow-convergence and poor spatial resolution issues inherent in some previous methods by implementing a deep ordinal regression network, along with a space-increasing discretisation strategy. Wofk *et al.* [18] propose a lightweight network suitable for embedded system applications, which is significantly faster than previous methods while retaining accuracy. Lee *et al.* [96] achieved state-of-the-art results by utilising novel local planar guidance layers located at multiple stages in the decoding phase.

Current state-of-the-art is achieved by Bhat *et al.* [97] who achieve their performance via a transformer-based architecture block that divides the depth range into bins whose center value is estimated adaptively per image. Final depth values are computed as linear combinations of the bin centers.

2.4.2 Semi-Supervised Depth Estimation

A common bottleneck in the development of a deep learning framework is the lack of training data with associated ground truth information. In the case of depth estimation the ground truth information is often obtained from additional sensors, which suffer from their own inaccuracies, which limit the effectiveness of the learning algorithm. To address this problem, several recent works have attempted to learn a depth map without the requirement of the learning algorithm to be only supervised by annotated ground truth information. Semi-supervised methods involve combining training data that includes ground truth annotation with training data that does not include ground truth information. The aim is to use the additional training data to improve upon supervised methods.

Garg *et al.* [98] first introduced an unsupervised deep learning framework for single-image depth prediction. In this work, an image pair with a small, known displacement between images is used. The predicted depth map is then inverse-warped using the known relative displacement to reconstruct the second image. Photometric error, rather than per-pixel depth displacement is then used as the supervision signal. Similarly, Godard *et al.* [99] also utilised the concept of training with an image pair. Epipolar constraints are utilised along with an image reconstruction loss to generate a disparity map between the image pair, from which a depth map can be obtained. Likewise, Zhou *et al.* [100] use a combination of camera pose estimation and photometric error to learn depth from pairs of images in a video sequence. The relative camera pose between frames is learnt by a separate network, after which an image can be warped to align with it's neighbour and a photometric loss can be applied.

2.4.3 Depth Refinement

Similar to object pose estimation, depth estimation does not necessarily need to be performed in a single step. Depth refinement is often proposed as a means to improve upon depth maps provided by other modalities, such as a stereo camera or time-of-flight camera [101, 102]. Recent learning-based approaches that focus on depth refinement include the work of Gidaris and Komodakis [103] and Knöbelreiter *et al.* [104]. Gidaris and Komodakis approach depth-map improvement as a multi-learning problem with three modules; error detection, pixel-wise label replacement, and refinement. Their approach is applied in an iterative framework wherein the the first two tasks 'hard' mistakes in the depth map, while the refinement task aims to improve predictions around areas with fine detail. In contrast, Knöbelreiter *et al.* learn to refine a disparity map between a stereo image pair. They achieve this by

proposing a partially parameterised cost function which can be unrolled for a fixed number of iterations. They obtain their network structure based on the number of iterations, and then apply their trained model in a single forward pass.

2.4.4 Benchmarks and Datasets

Popular datasets for depth estimation include NYU Depth V2 [19] and KITTI [105]. Popular metrics for this task include the Root Mean Squared Error (RMSE), and the Mean Absolute Error (MAE). RMSE is a popular, general purpose error that is widely used for depth estimation. Taking the log of the RMSE is also a popular metric. MAE is often chosen as it is less sensitive to outliers than RMSE. For RMSE and MAE a lower result indicates a better model.

2.5 Context of this work within the literature

Problems involving offline state estimation via regression or classification from a fixed-input are typically solved with a data-driven modelling approach such as a deep learning algorithm. Estimates provided by these algorithms can be refined further through the use of a separate neural network, as discussed above for pose refinement. This refinement is generally done without incorporating any knowledge-driven optimisation techniques. Conversely, problems involving online state estimation often utilise knowledge-driven modelling approaches involving numerical optimisation or filtering algorithms. Some recent work has explored the possible uses of deep learning within these problems, as discussed in Section 2.1.2. In general, such hybrid methods tend to replace some component of a classical algorithm, such as front-end feature extraction, with a neural network. There has been relatively little focus within these applications on the opposing problem, namely combining knowledge-driven techniques with data-driven techniques within offline estimation problems. This leaves room for the possibility that combining the capacity of learning with the rigour of optimisation algorithms may lead to improved results in such applications.

Recent work in this direction includes [106] and [107]. The goal of [106] is to learn an optimisation algorithm that can be used to train a network. The aim of learning an optimisation algorithm is similar to this work, although in this work the step size is selected manually and the gradient is learnt explicitely and explicitly with a CNN, whereas [106] learn a general update term in place of the step size and gradient with an LSTM. However, the application, an optimisation algorithm with which to train a

network, is very different to the application of this work; learning the gradient of an optimisation algorithm within which to apply a network iteratively.

The work by [107] involves learning a gradient descent scheme for iteratively applying a neural network to solve a problem. This scheme involves using a Recurrent Inference Machine [108] to learn the iterative framework. These works thus involve combining learning with optimisation strategies by attempting to learn the optimisation framework itself.

Most similar to the presented work is the work of [40], which implements a residual-based refinement scheme for optical flow estimation. This work builds upon previous optical flow literature, in which it is common to implement a coarse-to-fine refinement scheme via pyramid levels within one network, or by chaining multiple networks. [40] extend this work by demonstrating that equivalent or better results can be obtained with equivalent or less parameters by substituting the network chain for a single network with a single set of weights. Key differences to the presented work include that in this work supervision is performed on a per-iteration basis, while [40] combine the per-iteration loss into an overall loss. Unlike this work we also include an additional autoencoder architecture to inject previous estimate information.

The work discussed in this section either aims to learn the residual [40], or a general update term of an optimisation algorithm [106, 107]. However, a key hypothesis of this work is that there is merit in explicitly learning the state gradient, while using a separate step-size parameter to control the influence of the gradient on the iterative update. There appears to be limited prior research that investigates the utility of a learning a gradient that can be combined with a scaling parameter to solve general estimation problems. Such an approach, if implemented correctly, could be expected to combine the benefits of data-driven learning algorithms and knowledge-driven optimisation algorithms.

Chapter 3

A Robotic Harvester for Green Asparagus

This chapter aims to provide an overview of a robotic harvesting project targeting green asparagus that was undertaken with aims of commercialisation. The chapter begins with a summary of current growing and harvesting practices in Section 3.1. Next, an overview of the commercial potential of a green asparagus harvesting robot is provided in Section 3.2. Motivated by the insights obtained from this market research, a prototype robotic harvesting platform is presented in Section 3.3. This chapter ends with Section 3.4 which provides a discussion of the key learning outcomes of this robotic harvesting project.

3.1 Green Asparagus Growing and Harvesting

A recent survey by the Australian Department of Agriculture and Fisheries (DAF) found that labour is by far the most significant production/business cost for growers [109]. Labour related problems (e.g. increasing cost; coordination, managing and hiring large numbers of staff; dealing with personal issues; availability and access to skilled, trained, experienced and reliable people that do the job well; backpacker tax; etc.) were found to be the largest barrier to running a profitable business. For growers, reducing labour is integral to remain viable, competitive and sustainable, and enviable (the way of the future). In Australia, harvesting labour costs, on average, account for 30-40% of total production costs. As the harvesting costs for growers represent a significant portion of the total production expenditure, any efficiency gains in this area would provide a direct contribution to profitability. The major cost associated with harvesting is generally the cost of labour, particularly when harvesting is undertaken by hand, as is the case for crops that require selective harvesting.

Green asparagus is one of the few perennial crops that need multiple harvests

Figure 3.1: Manual harvesting of green asparagus [2].

throughout the yearly production. Harvest is usually daily because of the rapid growth of spears. Asparagus spears are usually harvested at 20-23cm lengths for fresh market use. Asparagus belongs to crops with very high total costs per hectare (ha) of growing area. Higher total costs to 1 ha of harvested area and relatively low yields per ha causes high product costs per tonne of asparagus ($\tilde{6}$,540 euro/ha) [110]. Labour costs make up 24-30% of total production costs. Types of labour costs include tractor/forklift driving, irrigation/fertilization, hand-harvest, pick-up, and packing (on farm). Harvesting is predominately by hand, with mechanical aids (e.g. motorized carts) being used to transport workers along/over rows. Manual harvesting is performed using a specifically designed harvesting knife, which is used to cut the spear either at ground level, or just beneath ground level. An example of this process is shown in Figure 3.1. As a result, the horticulture industry in Australia remains heavily reliant on manual labour, and is highly affected by labour costs.

Robotic harvesting offers an attractive potential solution to reducing labour costs while enabling more regular and selective harvesting, optimising crop quality and therefore profit. These potential benefits have spurred research in the use of agricultural robots (AgBots) for harvesting horticultural crops over the past 3 decades. Yet, harvesting robots are still far from mature, and harvesting is still predominately manual due to the limited performance of current robots. More so than other crops that require selective harvesting, such as capsicum, chilli, and lettuce, green asparagus offers an attractive target crop for robotic harvesting due to the lack of occluding foliage and simple harvest criteria. For these reasons a number of recent robotic harvesting platforms have been developed for green asparagus, however industrial

interest has been low due to reports of poor havesting success rates and/or high damage rates to spears and paddocks [60].

3.2 Commercial Potential

This project was undertaken with the long term aim of commercialising a robotic harvesting platform. It was therefore crucial to understand the market landscape that the end product would be part of. In this section the green asparagus and AgBot markets in Australia are summarised.

3.2.1 Green Asparagus Market

Research on green asparagus harvesting mechanisation/automation started in the early 1950s. However, labour costs were relatively low and there was limited interest in investing in machines [63]. In the 1990s, increasing costs of labour associated with lower labour availability contributed to restarting the research on mechanisation/automation of asparagus harvesting [63]. A recent survey (2016) by the Australian Department of Agriculture and Fisheries (DAF) found that labour is by far the most significant production/business cost for growers10. Reducing labour is integral (to remain viable, competitive and sustainable) and enviable (the way of the future) [109]. Driverless (autonomous) tractors and automated harvesting are currently the main priorities of growers – as while the cost per labour unit cannot be reduced, growers want to be able to do something about the number of labour units' required [109]. Asparagus fields can be harvested more than 20 times during the harvest season (multiple harvesting cycles to accommodate a range of maturity levels within the crop and harvesting at different lengths for different markets) and finding and keeping a labour force for the entire harvest is a challenge [110]. Mechanical/autonomous harvesting offers a solution to the sourcing of quality labour and removes 50-70% of the total costs of labour [111].

Due to poor harvesting speeds and/or low harvesting success rates, modern-day selective harvesting robots are non-economical for growers. Due to the articulation complexity required for 'pick and place' harvesting, cycle time (see-grasp-cut-place) for a single crop target is high (e.g. asparagus – GARotics - 2-5sec [63]). Although long cycle times are admissible for some controlled environments (e.g. green houses), these cycle times are impractical for many outdoor field crops – especially asparagus where daily harvesting must be completed by mid-to-late morning (as spears grow substantially during sunlight hours). A review (2013) of 50 harvesting robots (of

high value crops) found an average localisation success of 85% (meaning that 15% of the crop was not detected during harvesting) and an average harvesting success of 66% [54]. Yet a review (2008) of the economic comparison of selective mechanical versus manual harvesting of asparagus found that a harvest success rate of >85% (with a collateral damage rate of 5% and a damage to spears harvested rate of 5%) was needed for selective mechanical harvesting to be more profitable than manual harvesting [112]. The proposed platform and harvesting approach address the issues of a slow harvesting rate (an industrial fast delta robot combined with a cut and throw harvesting action – enabling a harvesting speed of <0.4 sec/spear) and a low harvesting success rate (state-of-the-art perception system with limited occlusion and lighting variations and higher resolution data – enabling a harvesting accuracy of >90%).

An increasing demand for asparagus in fresh food products is currently fuelling a boost in the global asparagus market [113]. During 2017-2027 it is predicted that over 10 million metric tonnes of asparagus will be consumed worldwide, resulting in revenues worth over $37 billion [113]. In terms of volume, this market is expected to display a 2.2% CAGR (Compound Annual Growth Rate), while global asparagus revenues are projected to grow at 3.1% CAGR, during the forecast period [113]. In terms of production, Asia is expected to remain the world's largest asparagus producer, followed Europe, North America, Australia and NZ, and Africa. Internationally Australia is about the 8th largest producer in terms of tonnage, with Victoria producing 95% of the national asparagus crop with a gross annual value of $47 million [114].

3.2.2 AgBot Market

The global AgBot market was valued at $2.75 billion(USD) in 2016 and is projected to reach $12.80 billion by 2022 at CAGR of 20.71% [115]. The most significant factor driving the AgBot market is the increasing focus on farm efficiency and productivity. Harvesting management is the most widely used application in agricultural robotic farming as it plays a vital role in high value crops and helps farmers and growers in maximising their yields [115]. Today most AgBots are in prototype or early stage of commercial trial phase [115]. The global agricultural equipment market was estimated to be $107.62 billion in 2017 and is projected to grow (CAGR = 5.31%) to reach $139.41 billion by 2023 [116]. Factors such as increasing mechanisation level in the agriculture industry, scarcity of skills labour and rising labour cost, government subsidies and growing need for operational efficiency and profitability are expected

to fuel this market over the next 5 years. Situated within this market, agricultural implements (specifically for harvesting) is expected to grow from $12.42 billion (2017) to $16.10 billion (2023) at a CAGR of 5.25% [116].

A recent (2016) review found that Australian growers are interested in the further application of automation, robotics and sensing technologies in field and shed operations [109]. The growers priorities (in order of importance) were automated crop health monitoring, autonomous weed management, autonomous all-purpose adaptable platforms, sensing and sensor networks, robotic harvesting, increased packing line efficiency, increased packing shed efficiency, managing vertebrate pests and virtual fencing [109]. Robotic harvesting was the main priority across all regions, as high impact but difficult, and was the top aspirational response in surveys, but it was seen as still some time off in the future and needing substantial R&D investment [109]. To address the growers' needs, AgBots are being developed locally to help Australian farms boost productivity (see: Table 3.1). However, most of these AgBots are targeting crop health monitoring and weed management, and all are several years away from being commercially viable.

At the time of writing, there are no examples of commercially successful selective harvesting robots for green asparagus, or for any other crop [117]. Key reasons for this include poor detection success rates and slow harvest cycles [63]. It also appears likely that there is a disconnect between robotic platforms developed in an academic environment, and the realities of commercialising this technology. Reasons for this include the gap between in-lab and on-farm performance of AgBots [54, 117], the need to integrate into existing growing systems without requiring a large overhaul [118], and the lack of funding for commercialisation of the technology [119, 118].

Despite the current lack of commercilisation success, interest in the development of such platforms is growing. This can be seen from a recent patent search (Figure 3.2) targeting green asparagus harvesting technology. There are currently 160 patents for an asparagus harvester, of which 47 are for a selective asparagus harvester, 34 are for a selective green asparagus harvester, 6 are for an asparagus harvesting robot, and 4 are for a green asparagus harvesting robot. The 4 patents for green asparagus harvesting robots were filed in 2016-2018.

Given the market potential and high interest, it appears that robotic harvesters, particularly for green asparagus, are likely to have a large impact on agricultural practices in the future.

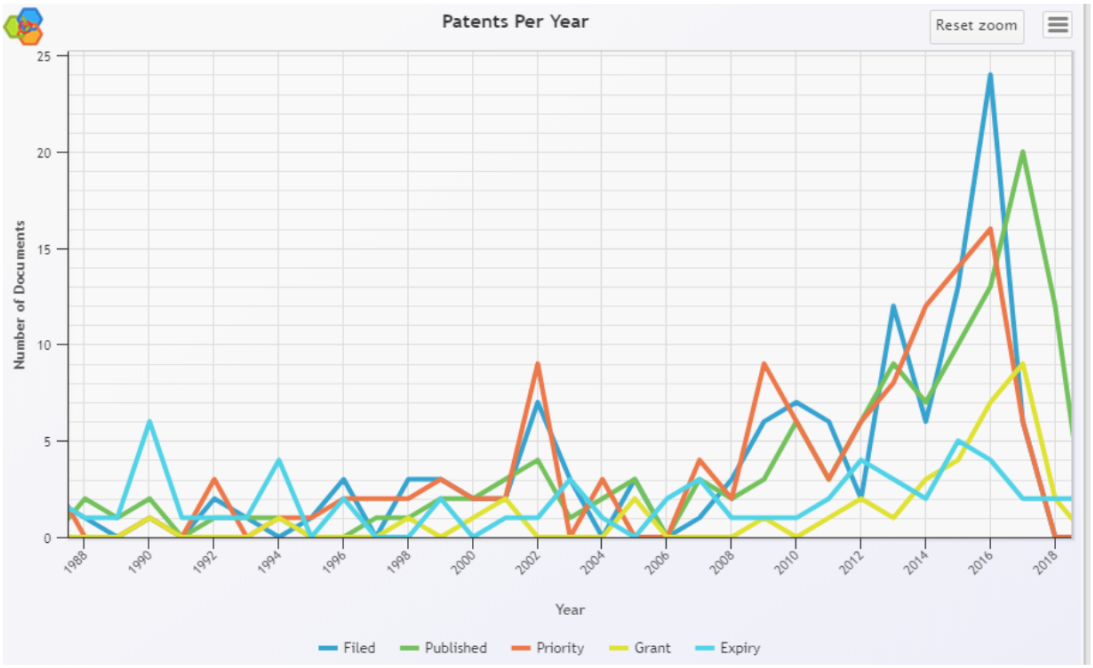


Figure 3.2: Patent search for asparagus harvesters.

3.3 A Robotic Harvesting Platform

The proposed robotic harvesting platform has 3 key systems; the Manipulation, Transportation, and Perception systems. This section provides an overview of these three sections, while Chapter 4 focuses on the algorithms used in the perception pipeline.

3.3.1 Manipulation System

Based on market research and previous harvesting robot performance it appears that a key roadblock to widespread adoption of this technology is harvesting speed. To be incorporated into existing farm procedures with minimal disruption the harvester needs to operate at speeds comparable to manual labour. However, it is common for robotic platforms to operate at much slower speeds, but to be operational at all hours. While this allows for large harvesting quantities despite slow harvest speeds, it requires an overhaul of existing farm procedures, such as warehousing and packing. This makes it difficult to undertake a simple trial of a robotic platform.

To address the speed issue the proposed manipulator system includes an industrial delta robot with an end-effector (i.e. harvesting tool) comprised of a simple glove structure with a fixed blade at the bottom. The Delta robot and end-effector

Robot	Function	Developer
Ladybird	A general-purpose research robot that conducts on-farm crop intelligence and crop manipulation activities	Australian Centre for Field Robotics
RIPPA	Real-time fruit detection. Has the ability to collect data using sensors that can map an area of crop and detect weeds as well as foreign objects	Australian Centre for Field Robotics
RIPPA with VIPPA	High-speed spot spraying of weeds. Autonomous real-time weeding and spraying of weeds, crops, or pests	Australian Centre for Field Robotics
SwagBot	Livestock monitoring and work. Can herd cattle and monitor the health of livestock. Can autonomously navigate predefined farm routes, detect and spray weeds and collect soil samples	Australian Centre for Field Robotics
Digital FarmHand	Able to analyse row crops and automate basic farming chores (weeding and seeding)	Australian Centre for Field Robotics
Mantis and Shrimp	Autonomous rovers that can produce 3D images that offer an overall picture of crop conditions. Working towards an automated pruning recommendation system	Australian Centre for Field Robotics
AgBot II	Can identify and spray weeds and can decide in real-time which weeds should be sprayed and which should be removed by mechanical or thermal methods	QUT, SIFR, DAF
Harvey	A harvester that can autonomously harvest sweet pepper in protected cropping environments	QUT, SIFR, DAF
Swarms (Swarm Robots)	Autonomous farming machines for spraying and weeding	SwarmFarm Robotics
-	Autonomous seeder for broad acre crops (a robotic seeding system)	University of NSW
-	Prototype robotic strawberry harvester	Magnificent Pty Ltd
FFRobot	Robot harvesting platform that emulates human hand picking for bruise-free fruit harvesting	FFRobotics
CAMIA	One-row, above ground asparagus harvester	CAMIA at University of Wollongong

Table 3.1: AgBots under development in Australia.

design are shown in Figure 3.3.

Delta robots are commonly used in industrial pick and place operations that require fast, repetitive movements. These robots can move as fast as 10m/s and accelerate as fast as 10m/s^2, which will allow the harvesting tool to travel across a 1m wide asparagus bed in less than 0.1 sec. The capabilities of the delta robot allow for a novel harvesting approach for asparagus, i.e. the end-effector has no moving parts and a 'cut and throw' (otherwise called a ballistic trajectory) harvesting action. An illustration of this harvesting approach is provided in Figure 3.4.

The reduction in mechanical complexity of the end-effector is expected to improve reliability within a farm environment, where dirt/water might otherwise corrode/-damage the harvesting tool. The 'cut and throw' approach creates minimal contact points between the crop and harvesting tool and the crop and the collection point. The throwing action will decrease crop contact points by relying on the momentum of the robot throughout the harvesting action to keep the harvested spear in contact with the harvesting tool. The harvested spear is then sent to the collection point via a throwing trajectory, such that the momentum is minimised at the collection point.

(a) Delta Robot [120].

(b) End Effector [1].

Figure 3.3: Manipulation System Hardware.

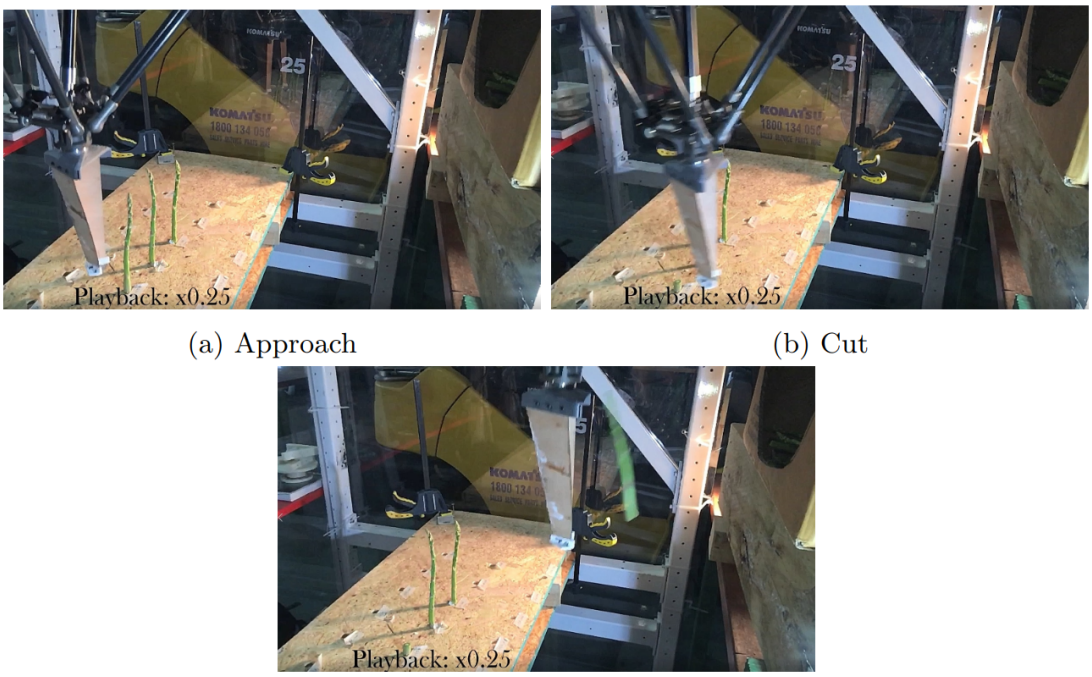
Two collection trays are used, one located on either side of end-effector, that can be filled with water to further prevent collection-point damage to the harvested spears. By implementing the throwing action, harvesting cycles can be significantly reduced (as compared to the traditional 'pick and place') due to the reduction in movement required by the robot. Essentially the end-effector will swing back and forth (in a figure-8 pattern) cutting in both directions.

3.3.2 Transportation System

The proposed transportation system involves integrating the perception and manipulator systems into a frame that can be mounted/hitched onto the back of a tractor (see: Figure 3.5). The frame will also incorporate a power generator (for the mechanics, electronics and computing) and will need to be designed in such a way that the robot's end-effector can move freely within the frame while still enabling tuning for suspension for traversing bumpy terrains. Ideally, this design will enable the harvester to be operated by one person using existing farm equipment. Ultimately, the platform will offer growers harvesting speed and accuracy (preventing wastage and loss) at a competitive cost (compared to manual labour) to run.

3.3.3 Perception System (Hardware)

The hardware for the perception system consists of three Chameleon 31S4 Point Grey cameras.The camera lenses are the Point Grey 3.5mm fixed focal lenses. This configuration was chosen based on the following list of selection criteria: industrial-level image quality, high resolution, global shutter, high frame rate, full colour, cost-effective,



(a) Approach (b) Cut

(c) Throw

Figure 3.4: Cut and throw (ballistic harvesting) [1].

synchronicity, programming interface. The three cameras will be arranged in a single horizontal plane, attached to the same horizontal bar of the enclosure. The cameras will be mounted and oriented such that their fields of view intersect with the expected location of the asparagus. The bar that the cameras will be mounted on will be located approximately two thirds of the way up the enclosure.

To keep lighting conditions consistent in the field the proposed perception system will be housed within a darkened enclosure with its own internal lighting. The lighting will consist of four COOLON Machine LED lights. One light will be situated on each of the four bars that form the perimeter of the enclosure roof. To create more uniform light, the lights will be pointed away from the asparagus spears to diffuse the light throughout the enclosure. This will enable day-time and night-time harvesting. An early version of the proposed perception system hardware (with extra lights, camera, and without a darkening cover) is shown in Figure 3.6.

3.4 Project Termination - Lessons Learnt

The aim of this project initially involved the development of a prototype robotic harvester capable of selectively harvesting green asparagus. The project began after

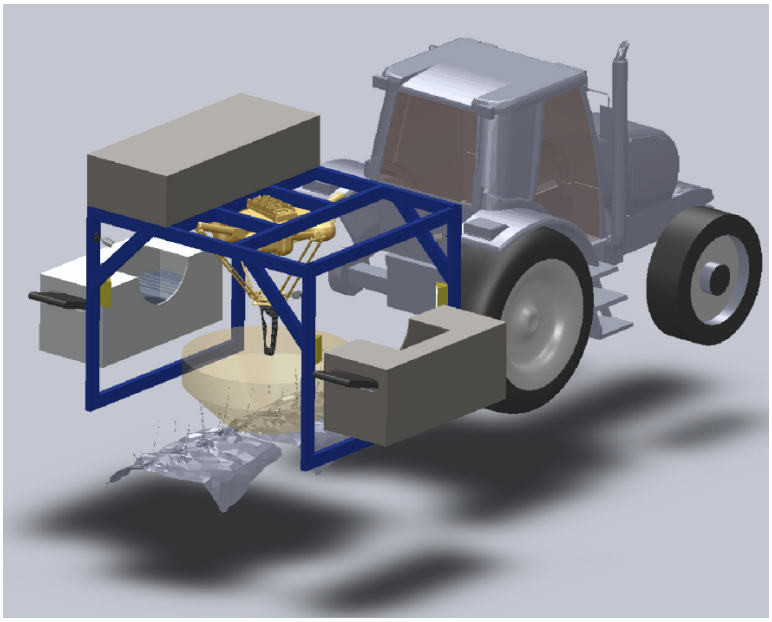


Figure 3.5: Proposed Robotic Harvesting Platform (image courtesy of Alex Martin).

ANU won a tender process with a local asparagus farm, with initial funding obtain internally. However, this funding was limited, leading to the initial stages of the project being conducted within a number of distinct research projects, each focusing on a different aspect of the harvester. This initial work involved conducting proof-of-concept work in-lab, largely related to the perception and manipulation systems. This work was conducted to test the following assumptions that were core to the proposed harvester design

- the harvester must operate at speeds comparable to human labour to enable smooth adoption of the new technology
- high speed harvesting can be obtained by implementing a cut-and-throw harvesting method
- simple perception algorithms could be implemented to obtain a favourable balance between accuracy and speed
- existing farm technology could be utilised to transport the platform

As the project progressed it became clear that without a larger team consisting of additional engineering support, development of a complete prototype harvester was infeasible. Without additional funding a team could not be assembled to develop the prototype. Without a prototype additional funding appeared unavailable. For this reason the project ultimately stalled.

Figure 3.6: Early Version of Vision System Hardware Setup. The proposed system contains 3 cameras and 4 lights inside a darkened enclosure.

This experience led us to conclude

- growers wish to see successful proof-of-concept on-farm before committing to a project
- priority must be given to developing a robust platform capable of functioning in a range of environmental conditions
- funding for large-scale development of commercial robotic technology by academic institutions is considered high-risk by some industry groups due to a perceived misalignment of goals
- the commericalisation pathway should be clear before significant resources are devoted to a industry-focused project

3.5 Conclusion

This chapter presents an overview of green asparagus growing and harvesting practices, and outlines the attributes that make it an attractive target crop for robotic harvesting. The commercial potential of a robotic harvester for green asparagus in Australia is investigated. It was found that current robotic harvesters are generally not economical for growers due to low harvesting success rates and slow cycle times. These insights led to the development of a prototype robotic harvesting platform, which is also presented in this chapter. The harvester was developed for easy integration into existing farming systems, and for high-speed harvesting. A lack of funding eventually led to the termination of this project, and the lessons learnt from this process are also discussed. Of these lessons, a key takeaway is the need to have a clear commercialisation strategy in place before significant resources are committed.

Chapter 4

Perception Pipeline for a Robotic Harvester of Green Asparagus

In this chapter a prototype perception pipeline for detecting, localising, and tracking green asparagus spears is presented. Specifically, the height, and location of each spear relative to the ground plane is tracked. This is the target 'state' for this problem, as it encodes all information necessary to selectively harvest asparagus spears. Initial in-lab and on-farm results of the pipeline are presented.

4.1 Algorithm Architecture

The algorithm architecture is split into two components that operate in parallel. These components are titled Extrinsic Camera Calibration, and Perception Pipeline, and are discussed in Sections 4.1.1 and 4.1.2 respectively.

4.1.1 Extrinsic Camera Calibration

Extrinsic camera calibration is computed online as a separate process from the perception pipeline. It is computed at a low frequency and used to update camera pose and ground plane estimation periodically. This ensures that the cameras remain well calibrated throughout long periods of on farm use even in rough conditions. Extrinsic calibration involves computing a camera pose matrix $T \in \mathbf{SE}(3)$ for each camera, where $\mathbf{SE}(3)$ denotes the Special Euclidean Group. The calibration is implemented in the open-source OpenMVG framework [121] via a Structure from Motion (SfM) algorithm computed from sparse feature points that are matched between frames [121][1]. OpenMVG (Multiple View Geometry) is a C++ open-source library for computer-

[1]Code available at: https://github.com/kennege/openMVG/tree/develop_multi_camera_calibration

Feature Detector	Feature Descriptor	Feature Descriptor Software Package
SIFT	SIFT	OpenMVG
AKAZE	Modified SURF	OpenMVG
AKAZE	Modified Local Difference Binary (MLDB)	OpenMVG
AKAZE	Local Intensity Order Pattern (LIOP)	OpenMVG
SIFT	SIFT	OpenCV
AKAZE	AKAZE	OpenCV
SURF	SURF	OpenCV
ORB	ORB	OpenCV
BRISK	BRISK	OpenCV

Table 4.1: Feature Detectors and Descriptors Tested for this Project.

vision scientists targeted for the Multiple View Geometry community [121]. It provides ready-to-use tools and Structure from Motion (SfM) pipelines along with other tools for localization and geodesy.

Testing of various types of features was implemented within the OpenMVG framework. The feature detectors and descriptors that were tested are summarised in Table 4.1.

For each of the implemented feature detector/descriptor combinations shown in Table 4.1 the number of detected features, number of feature matches, matches/feature, and implementation time were evaluated using images obtained on an asparagus farm. Results are included in Figures 4.1,4.2,4.3, and 4.4. Based on this evaluation, the OpenCV implementation of the AKAZE feature detector and descriptor was chosen. Figure 4.5 provides an example of the feature matches obtained using the OpenCV implementation of AKAZE.

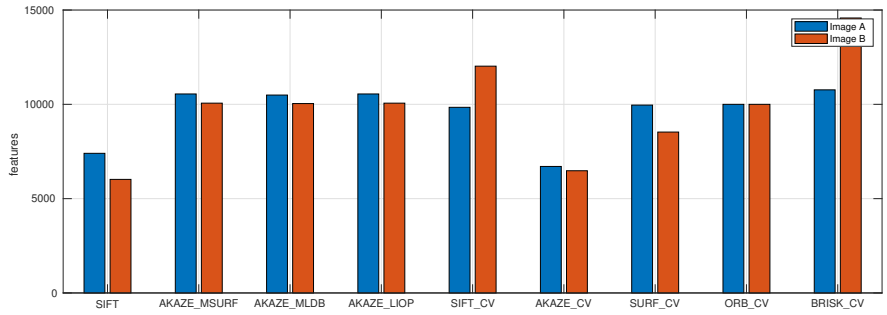


Figure 4.1: Number of detected features per image.

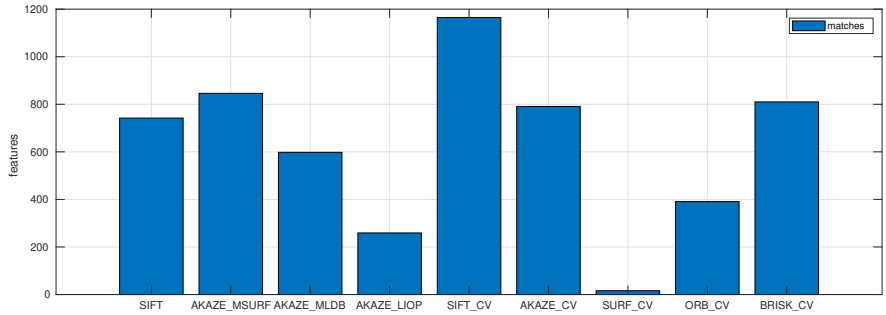


Figure 4.2: Number of matches per image pair.

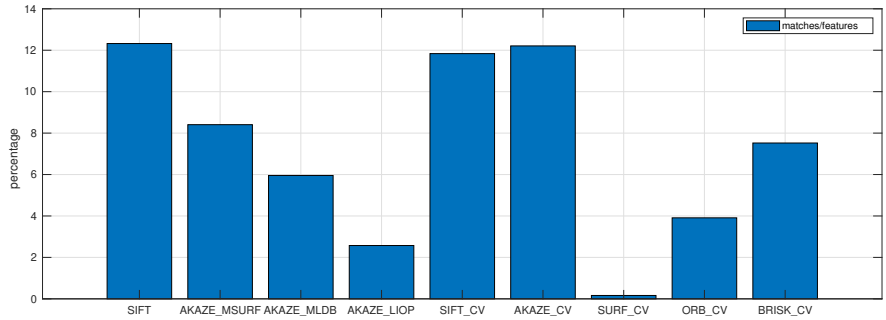


Figure 4.3: Percentage matches per feature.

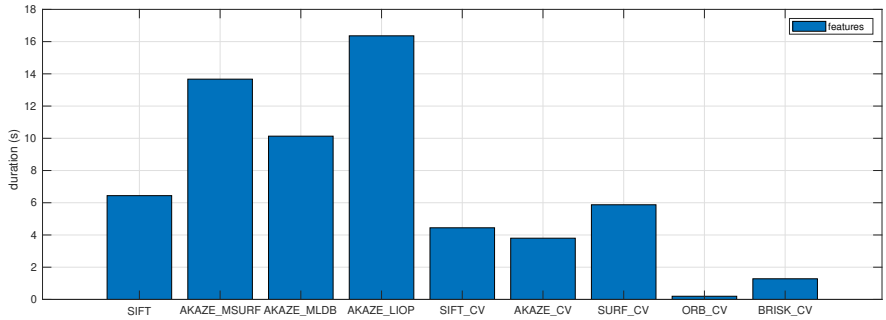


Figure 4.4: Time taken for feature detection and description.

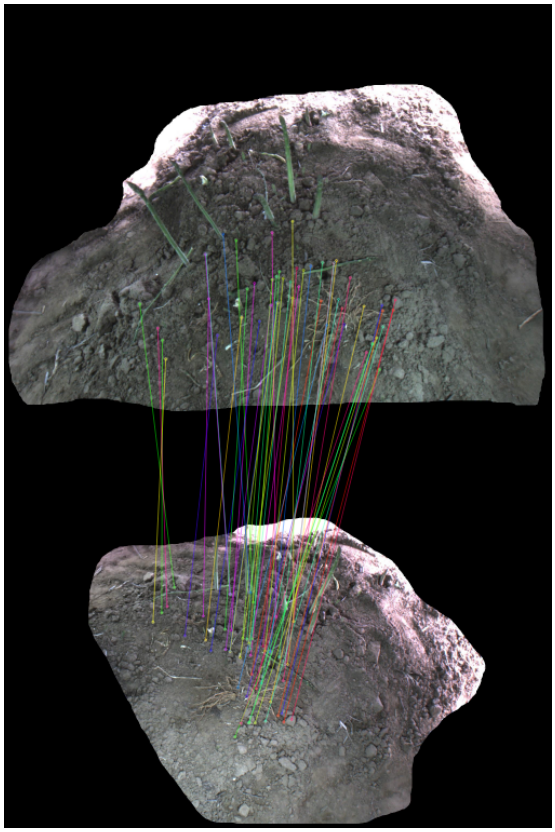


Figure 4.5: Feature matches obtained using the OpenCV implementation of AKAZE.

Aside from feature detectors and descriptors, different geometric models were also trialed. The geometric models that were considered were the Homography, Essential, and Fundamental matrices. Each model has different characteristics, and requires different information to provide an estimation. For example, the Homopgraphy matrix assumes that the cameras who's geometric relationship is being modelled can both view the same planar surface. Each model was evaluated to determine how well it matched the provided feature and camera location information. This was measured as the residual mean. The models were estimated for each time step (one image per camera per time step), and the models were also evaluated based on how well the estimate at one timestep generalised to other timesteps (where the features would be different but the relative camera positions and orientations should be very similar). This is referred to as 'guided matching'. This information was obtained for a large range of images captured on an asparagus farm, and some key results are included in Tables 4.2 and 4.3.

Geometric Model	Num 3D Points	Residual Mean
Homography Matrix	2098	0.218
Essential Matrix	13,097	0.385
Fundamental Matrix	14,302	0.386

Table 4.2: 3D reconstruction for different geometric models.

Geometric Model	Inliers	Mean Reprojection Error	Guided Matches	Inliers	Updated Mean Reprojection Error
Homography Matrix	209	2.15	205	131	2.19
Essential Matrix	640	1.18	1474	1291	1.19
Fundamental Matrix	663	1.25	1405	1320	1.07

Table 4.3: Guided matching using different geometric models.

From the analysis presented in Tables 4.2 and 4.3 it was determined that the Essential and Fundamental matrices provide the best geometric model estimation. Of these two options the Essential matrix was chosen, as this model can utilise intrinsic camera information, which is computed for each camera prior to harvesting.

Based on these design choices, the calibration algorithm presented in Algorithm 1 was developed.

Algorithm 1 Extrinsic Camera Calibration

Input N // number of cameras
Input $K\_\{1, ..., N\}$ // intrinsic matrices
Input $I\_\{(1, ..., N), (1, ..., n)\}$ // n images for N cameras
for $i = 0; i < n; i++$ **do** // for number of images per camera
    if i==0 **then**
        nPairs = (N(N-1))/2 // only match spatially
    else
        nPairs = (2N(2N-1))/2 // match spatially and temporally to next timestep
    for j=0;j<nPairs;j++ **do**
        if image hasnt been seen before **then**
            Perform contrast-limited adaptive histogram equalisation (CLAHE)
            Extract 200 best OpenCV AKAZE features
        if i==0 **then**
            Match features using Brute Force L2 matching
            Estimate Essential matrix
        else
            Match features using guided matching
        Perform Bundle Adjustment
        Update camera poses based on Bundle Adjustment

Views	Tracks	Residuals	Initial RMSE	Final RMSE	Time (s)
3	600	1800	0.339	0.122	0.053

Table 4.4: Example extrinsic camera calibration results from three images captured at a single timestep.

Table 4.4 provides an example of the calibration result for a single timestep. In Table 4.4, 'tracks' refers to number of image features, 'residuals' refers to the residual error of 3D point reprojections, 'RMSE' refers to root mean-squared error, and 'Initial/Final' refers to before/after the Bundle Adjustment step is applied.

4.1.1.1 Ground Plane Estimation

Given that the vast majority of matched feature points occur on the ground plane, this plane can be found via simple 3D plane fitting. The equation for the ground plane is defined as

$$ax + by + c = z,$$

for 3D points $(x, y, z) \in \mathbb{R}^3$. The constants a, b, c are solved for using

$$\begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ & \vdots & \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_n \end{bmatrix},$$

which can be represented as an over determined linear system $Ax = B$ and solved via the left pseudo inverse

$$x = (A^T A)^{-1} A^T B.$$

This ground plane provides a convenient 2D representation of the region of interest. The output of the extrinsic calibration is a matrix T representing the pose of each camera, and a ground plane G, all with respect to a common inertial frame.

4.1.2 Perception Pipeline

The perception pipeline incorporates the software for receiving images continuously from the multi-camera system, and information from the extrinsic calibration, and returning locations of asparagus to be harvested. Broadly, the procedures involved

in the pipeline for each time step are: generate a 2D probability map for each image, project these maps to a common ground plane computed from the extrinsic calibration, and detect potential asparagus 'hypothesis' on this ground plane. Finally, the hypothesis are filtered temporally across time steps to improve accuracy. The four **procedures** are discussed further in the following four subsections respectively.

Algorithm 2 Perception Pipeline

Input images from time $t = 0 : end$
Initialise Kalman Filter
for ($\mathbf{t} = 0 : end$) **do** // *for images each time step*
    for ($\mathbf{I}_{i=0:2}$) **do** // *for images this timestep*
        procedure 1. 2D Probability Map($\mathbf{I}_i$)
            $\mathbf{I}_i$ = yCBCR($\mathbf{I}_i$) // *colour space transform*
            $\mathbf{I}_i = \gamma(\mathbf{I}_i)$ // *perform Gamma correction*
            $\mathbf{p}_{(0,1,2),i}$ = PCA($\mathbf{I}_i$)
        procedure 2. Ground Projection($\mathbf{p}_{1,i}$)
            $\mathbf{P}_i = \mathbf{K}_i\mathbf{T}_G\mathbf{T}_i^{-1}$ // *projection matrix*
            $\mathbf{H}_i$ = $\mathbf{P}_i$[:][0,1,3]
            $\mathbf{G}_i$ = warp($\mathbf{p}_{1,i}$,$\mathbf{H}_i$) // *backward projection*
    procedure 3. Find Hypothesis($\mathbf{G}_{i=0:2}$)
        $\mathbf{G} = \sum_{i=0}^{2} \mathbf{G}_i$ // combine images
        $\mathbf{G}$ = thresh($\mathbf{G}$)
        $\mathbf{G}$ = removeClutter($\mathbf{G}$)
        $\mathbf{L}_{1:n}$ = fitLines($\mathbf{G}$) // fit lines 0:n
        $\mu_{1:m}, \Sigma_{1:m}$ = hypothesis($\mathbf{L}_{i=1:n}$,$\mathbf{L}_{j=1:n}$,$\mathbf{L}_{k=1:n}$)
    procedure 4. Filter Hypothesis($\mu_{1:m}, \Sigma_{1:m}$)
        $\mu_{1:m}, \Sigma_{1:m}$ = KalmanFilter($\mu_{1:m}, \Sigma_{1:m}$)
        for h=0:m **do** // for each hypothesis h
            if ($m_h >$ thresh & 20cm $< l_h <$ 25cm) **then**
                Mark h for harvesting

4.1.2.1 2D Probability Map

For each timestep images from each camera are first processed to produce a representation of probability of asparagus. The proposed approach involves colour space conversion from RGB to yCBCR, gamma correction, and Principle Component Analysis (PCA). The yCBCR colour space has been chosen empirically based on the relative contrast of plant matter to background. Gamma correction was computed with $\gamma = 1.5$. The first three principal axis are generated, and the second axis is used as the output probability map.

4.1.2.2 Ground Plane Projection

The known camera poses and ground plane provided by the extrinsic calibration are used to project each image plane I to the common ground plane, labeled G (see Figure 4.8). This is done by first computing the camera projection matrix $P \in \mathbb{R}^{3\times4}$ using

$$P = KT_G T^{-1},$$

where K is the intrinsic camera matrix computed offline via the Matlab Camera Calibration app [122]. The camera pose $T \in \mathbf{SE}(3)$ is obtained from the extrinsic calibration, and $T_G \in \mathbf{SE}(3)$ is chosen to transform the inertial frame of the extrinsic calibration such that it is aligned with the z-axis orthogonal to G. Let $P = [p_1, p_2, p_3, p_4]$ with each $p_i \in \mathbb{R}^3$ representing a column of P. The first three columns p_1, p_2, p_3 represent the vanishing points in G with respect to the x, y, and z axis respectively, in homogeneous image coordinates. As T_G is defined such that the z-axis of the new inertial frame is orthogonal to the ground plane then $p_3 \approx (0, 0, 1)^\top$, where $\approx$ denotes equality up to scale. Set $H = [p_1, p_2, p_4] \in \mathbb{R}^{3\times3}$. This is an homography between G and I. Points in each image frame I can thus be mapped to G using a backward homography projection,

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \approx H \begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix},$$

for $i \in I$ and $g \in G$.

An example ground plane image is provided in Figure 4.6. This image was created using the scale 1 pixel = 1mm. In this figure, each asparagus is represented by a 'chicken-foot' object corresponding to the same asparagus projected from three different views. This is a novel method of encoding multi-camera information in a single image plane. It is likely that the extra information encoded in this image (three views of each object in one image) could allow machine learning algorithms to extract high quality estimates of object location and pose, compared to typical single image-based algorithms that perform estimates from single RGB images.

4.1.2.3 Hypothesis Generation

The ground plane image is thresholded and morphological operations (closing with the OpenCV line structuring element [123]) are applied to remove 'clutter'. Lines

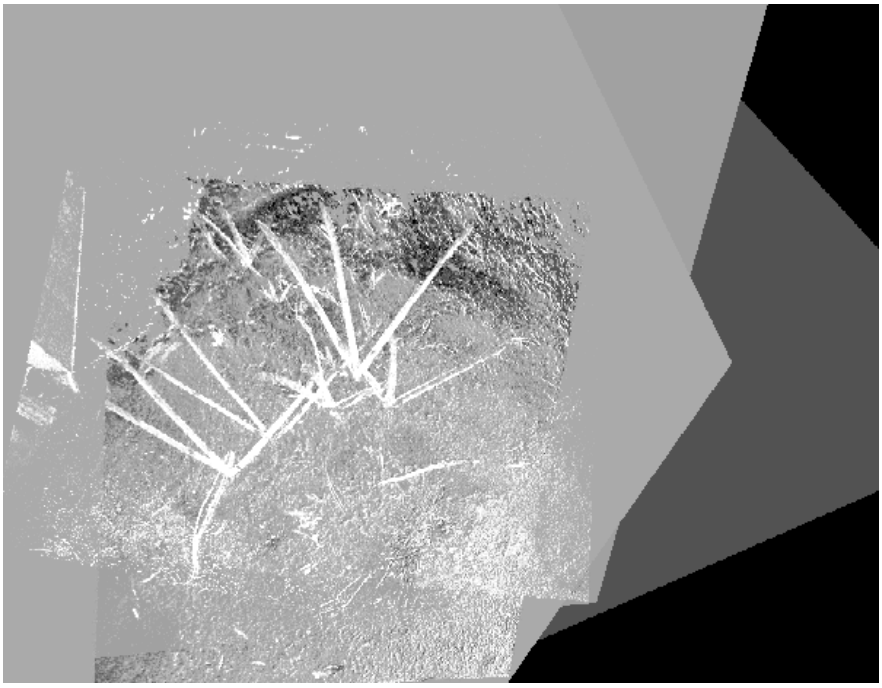


Figure 4.6: Ground Plane image obtained by projecting the 2D probability map from each camera view to a common ground plane.

are fit to each remaining morphological 'blob' via a least-squares approach. Area A, centroid c, and eccentricity e are also computed for each blob. All line intersections are computed, and a hypothesis of an asparagus 2D location within the ground plane is represented by every set of three intersections. The hypothesis mean μ_h is computed as the centroid of the triangle spanned by the three intersections,

$$\mu_h = \frac{1}{3}(a + b + c), \tag{4.1}$$

where $a, b, c \in \mathbb{R}^2$ represent the three intersections of the given hypothesis. The hypothesis covariance Σ_h is represented by the Steiner circumellipse of the triangle (see Figure 4.7) [124] and the score of each triangle edge, via

$$\Sigma_h = \begin{bmatrix} \frac{e}{3}\sum_{i=0}^{2} S_i & 0 \\ 0 & \frac{f}{3}\sum_{i=0}^{2} S_i \end{bmatrix}, \tag{4.2}$$

where e and f refer to the major and minor axis of the Steiner circumellipse respectively and S_i refers to a score of triangle side i.

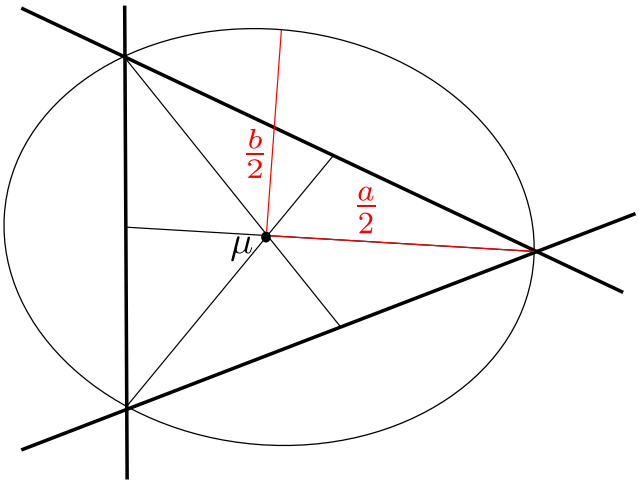


Figure 4.7: Steiner Circumellipse.

The following likelihood score S is proposed:

$$S = \left| 1 - \frac{e + \frac{A}{\mu_A}}{2} \right|,$$

where A and e are the area and eccentricity of the current asparagus 'blob' respectively and μ_A is found empirically to be the area of a typical asparagus 'blob' in pixels. This score is designed to provide a normalised measure of how well the eccentricity and area of the detected blob matches a typical asparagus. A hypothesis is added to the system state if it satisfies the condition

$$\left(\frac{1}{3} \sum_{i=0}^{2} S_i < 0.6 \right) \wedge \left(t < 1 \right) \wedge \left(\mu_h \in I_b \right) \wedge \left(||\Sigma_h|| < 1 \right),$$

where $t = |\frac{\bar{l} - \mu_l}{\sigma_l / \sqrt{3}}|$, $||x||_2$ is the L2-norm of x, and

$$\bar{l} = \sum_{i=0}^{2} 2||\mu_h - c_i||_2 \tag{4.3}$$

is the average length of the three asparagus 'blobs' in the hypothesis (in pixels), μ_l and σ_l are found empirically to be the mean and standard deviation in asparagus 'blob' length, c is the centroid of the current asparagus 'blob', and $x \in I_b$ indicates that x lies within the ground plane image, on an asparagus 'blob'. This condition was chosen empirically based on a range of tests run on farm and in-lab data. Asparagus hypothesis height in meters ($l \in \mathbb{R}$) can be obtained using Equations (4.1) and (4.3)

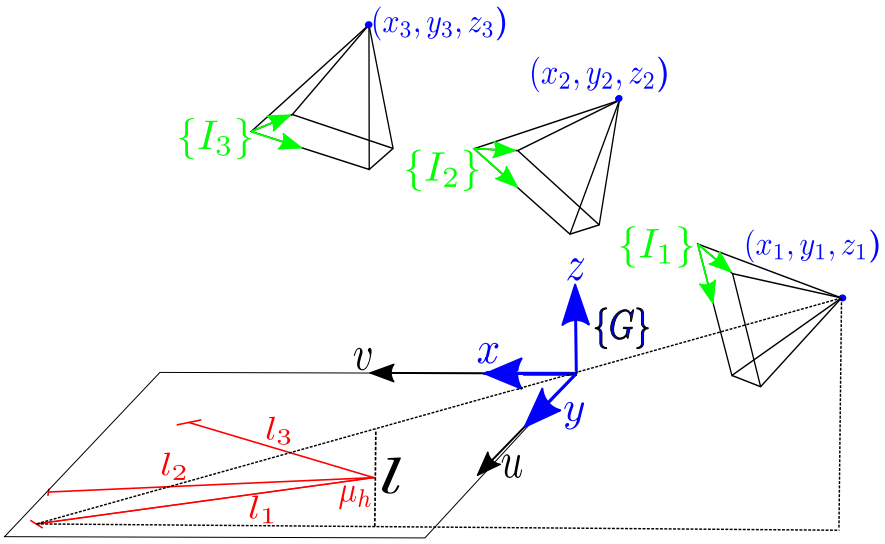


Figure 4.8: Relationship between image frames I_i and ground plane G. Where l_i is the 2D length of the asparagus in G projected from image plane I_i, and μ_h is the 2D position estimate and l is the height estimate.

and

$$l = \frac{1}{3}\sum_{i=0}^{2} \frac{\frac{l_i}{1000} z_i}{\frac{\sqrt{\mu_h^2}}{1000} + \sqrt{x_i^2 + y_i^2} + \frac{l_i}{1000}}, \tag{4.4}$$

where x_i, y_i, z_i represent the position of camera i with respect to G, and the scale factor of $\frac{1}{1000}$ is used due to the imposed scale of 1mm = 1 pixel in the ground plane image.

Using this formulation, the probability encoded in the 2D map is propagated to morphological blobs, then to lines via the line scores S, then to 2D location hypothesis via μ_h and Σ_h. Finally, these hypothesis are filtered temporally with a Kalman Filter.

4.1.3 Kalman Filtering

A 2D Kalman Filter (KF) is used to track hypothesis temporally through successive ground plane images. Given the expected harvester speed each asparagus spear will be in view for 30-50 frames. Temporal filtering allows hypothesis to be refined and pruned over this period.

The KF is applied to the following process model:

$$X_k = X_{k-1} + U_k + W_k,$$

with observations of the form:

$$Z_k = X_k + V_k,$$

where

$$X_k = \begin{pmatrix} x_k \\ y_k \\ l_k \end{pmatrix},\ U_k = \begin{pmatrix} u_k \\ v_k \\ 0 \end{pmatrix},$$
$$W_k \sim \mathcal{N}(0, Q_k),\ V_k \sim \mathcal{N}(0, R_k),$$
$$Q_k = \begin{bmatrix} \Sigma_{u_k} & 0 & 0 \\ 0 & \Sigma_{v_k} & 0 \\ 0 & 0 & 1 \end{bmatrix},\ R_k = \begin{bmatrix} \Sigma_{h,k} & 0 \\ 0 & \Sigma_{l,k} \end{bmatrix},$$

where k denotes timestep, x, y represent the 2D location of an hypothesis in ground plane coordinates, with associated measurements provided by μ_h from Equation (4.1), l represents the height of the asparagus hypothesis obtained in pixels using Equation (4.3) and converted to meters using Equation (4.4). The covariance Σ_h is found using Equation (4.2), and Σ_l represents the covariance of the height estimate. The covariances Q_k and R_k represent process and measurement noise respectively. Data association of asparagus hypothesis between frames is obtained via Lucas-Kanade Optical Flow (LKOF). Estimates for u_k, v_k, Σ_{u_k} and Σ_{v_k} are obtained empirically from the pixel-wise displacement obtained from the LKOF.

An asparagus hypothesis is considered to be valid if it is tracked for a set number of frames (referred to as 'track length' and labeled m_h in Algorithm 2). If a hypothesis is validated and its height meets the threshold condition it is marked for cutting. The 3D cutting point is fixed at 2cm above the ground plane. Figure 4.9 provides an illustration of the final result of the perception pipeline; a sparse point cloud computed via OpenMVG for extrinsic calibration combined with asparagus hypthesis obtained via homography projection and line-fitting.

4.2 Experiments

4.2.1 On-farm

Image data has been captured at a nearby asparagus farm on two occasions (Figure 4.10). This has allowed for testing of the performance of the physical perception system, and robustness of the perception pipeline to outdoor conditions.

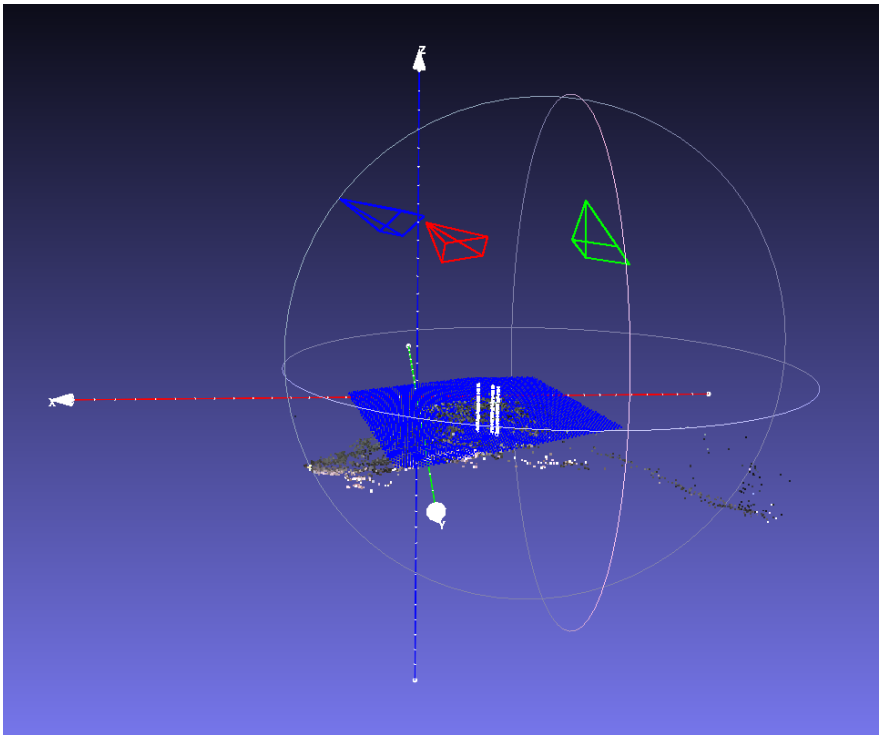


Figure 4.9: Output of perception pipeline for a single timestep.

Figure 4.12 presents an example result of hypothesis generation and filtering using on-farm data. While initial hypothesis detections are noisy, the KF reliably filters erroneous detections via an upper limit on the state covariance. Initial small covariance values are due to the size of the Steiner circumellipse. Covariances grow due to process noise, and when an hypothesis is not tracked between frames, and covariances decrease when a data assocation is made. During harvesting, an hypothesis will be considered valid if it has a certain track length. The on-farm dataset contains 30 frames, and the required track length was set to $m_h = 3$. Using this track length, the three hypothesis shown in Figure 4.12(h) were validated. At this point the asparagus height estimate will be obtained from the KF and harvested if it is tall enough. The 3 tall spears met the target track length ($m_h = 3$) after 8 frames and were marked for harvesting. The 2 smaller targets and a horizontal spear are not detected, but these would not meet the harvest threshold. No other locations surpassed the target track length for the remainder of the dataset (30 frames).

Figure 4.10: On-farm day and night experiments.

4.2.2 In-lab

Datasets simulating the perception system moving over an asparagus bed have been captured in-lab via the fabricated rig shown in Figure 4.11. A sliding bar attached to a motor system allows the camera system to move over an asparagus bed at the expected speed of the harvester. Using this rig, the results shown in Table 4.5 were generated.

Table 4.5 contains the results of six experiments, each with a different configuration of asparagus spears. Example configurations are shown in Figure 4.13. A dense 3D reconstruction computed in OpenMVG is used as a pseudo ground truth in this experiment. RMSE (mm) is the root mean squared error in the KF estimate, ξ_k for a given hypothesis, taken with respect to the estimate provided by the 3D reconstruction, ζ_k. That is

$$\text{RMSE} = \sqrt{1/N \sum (\xi_k - \zeta_k)^2}.$$

Exp	RMSE (mm)
1	0.137
2	1.829
3	0.128
4	1.778
5	0.207
6	0.333
Overall	**0.735**

Table 4.5: Results of In-lab Experiments. **RMSE (mm)** is the root mean squared error in the KF estimate.

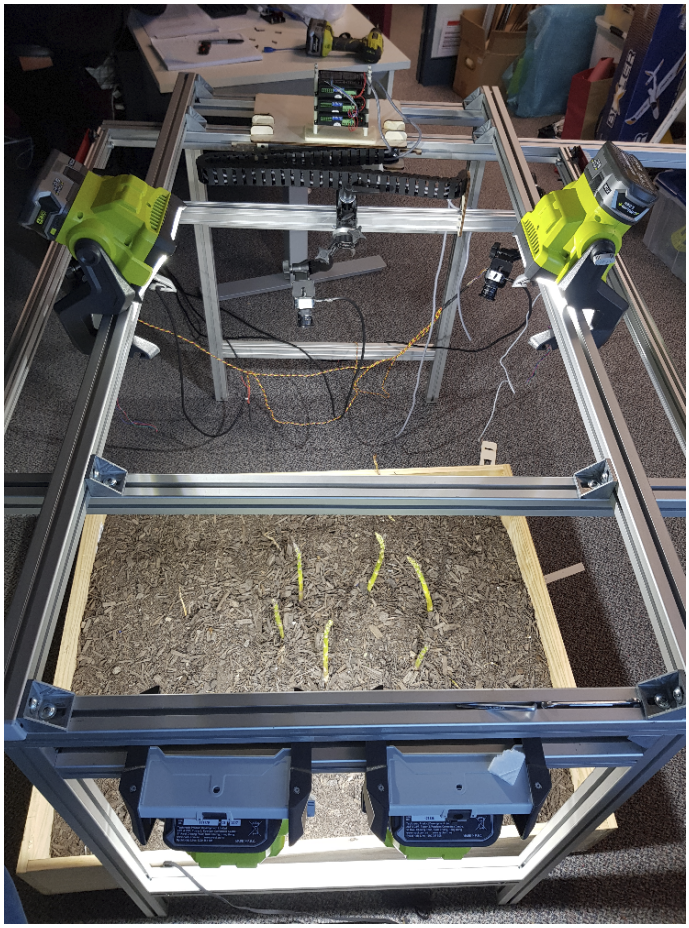
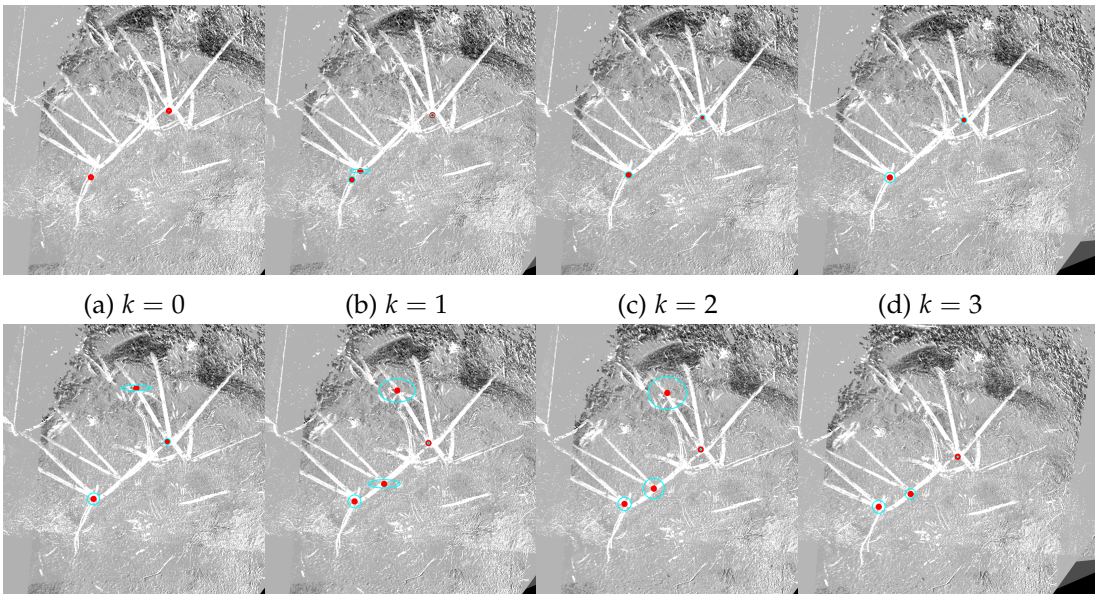


Figure 4.11: In-lab rig for harvesting simulation. A sliding bar attached to a motor system allows the camera system to move over an asparagus bed at the expected speed of the harvester.

Each dataset in Experiments 1-6 comprises of $\sim$ 30 time steps, and the required track length is once again set to $m_h = 3$. No horizontal spears were detected during the experiments. This is due to the ground plane projection, from which horizontal spears do not produce the 'chicken-foot' representation of vertical asparagus. This is advantageous as only upright spears are targeted for harvest. This is experiment indicates that the proposed pipeline finds asparagus cutting points robustly with a high level of accuracy.



(a) $k = 0$ (b) $k = 1$ (c) $k = 2$ (d) $k = 3$

(e) $k = 4$ (f) $k = 5$ (g) $k = 6$ (h) $k = 7$

Figure 4.12: Kalman filtering of hypothesis using the first 8 frames of on-farm data. The 3 tall spears met the target track length ($m_h = 3$) after 8 frames and were marked for harvesting. The 2 smaller targets are not detected, but these would not meet the harvest threshold. A horizontal spear is also not detected. No other locations surpassed the target track length for the remainder of the dataset (30 frames). Covariance ellipses increase between frames if the hypothesis is not detected, and decrease if the hypothesis is detected.

(a) Exp **2** (b) Exp **3** (c) Exp **6**

Figure 4.13: In-lab Experiments. Only upright asparagus should be targeted for harvesting.

4.3 Conclusion

This chapter presents a perception pipeline for robotic harvesting of green asparagus. The pipeline utilises simple algorithms and a multi-camera rig to obtain estimates of asparagus location in real-world conditions. A novel single-view ground plane projection is presented as a novel approach to condensing multi-camera information into a single image. When compared to results obtained from a dense 3D reconstruction algorithm, the proposed approach located asparagus spears with a mean relative error if 0.74mm. This indicates that the proposed approach yeilds comparable results to high accuracy, low speed approaches. Qualitative results indicate that the proposed approach also allows spears to be detected and tracked in on-farm conditions.

Chapter 5

Iterative Optimisation with an Innovation CNN for Pose Refinement

Object pose estimation from a single RGB image is a challenging problem due to variable lighting conditions and viewpoint changes. The most accurate pose estimation networks implement pose refinement via reprojection of a known, textured 3D model, however, such methods cannot be applied without high quality 3D models of the observed objects. In this chapter an approach, namely an Innovation CNN, to object pose estimation refinement is proposed. This approach is general and overcomes the requirement for reprojecting a textured 3D model to inject the initial estimate into the network. The approach improves initial pose estimation progressively by applying the Innovation CNN iteratively in a stochastic gradient descent (SGD) framework. Popular LINEMOD and Occlusion LINEMOD datasets are used to evaluate the method, and improvements of 4.2% and 8.12% are respectively obtained.

5.1 Proposed Innovation CNN

In this section a formulation for a general state estimation algorithm is presented and applied to the problem of object pose estimation. The network architecture, training loss, and network implementation are also discussed.

5.1.1 State Estimation

Motivated by filtering and optimisation principles [125], I treat object pose estimation as an offline state estimation task. In this context a *state* is an internal representation of a system that is sufficient to fully define the future evolution of all system variables.

A *state estimation* algorithm is a dynamical system that takes measurements from the true system as inputs and whose state is an estimate of the true system state.

Consider a dynamical system in a state $\boldsymbol{X}$ defined by

$$\dot{\mathbf{X}}(t) = f(\mathbf{X}(t), \mathbf{V}(t)), \tag{5.1}$$

$$\mathbf{y}(t) = h(\mathbf{X}(t)), \tag{5.2}$$

where $f(.)$ is the state model $\mathbf{V}(t)$ is an measured input signal, $h : \boldsymbol{X}(t) \rightarrow \mathbf{y}(t)$ is an output map, and t represents the current time. The output $\mathbf{y}(t)$ encodes the specific target information that the engineer is interested in, while the state $\mathbf{X}$ encodes the full information necessary for the propagation of the dynamical system.

The class of estimation algorithm considered in this paper are of the form

$$\dot{\widehat{\mathbf{X}}}(t) = f(\widehat{\mathbf{X}}(t), \mathbf{V}(t)) - \Delta(t), \tag{5.3}$$

$$\widehat{\mathbf{y}}(t) = h(\widehat{\mathbf{X}}(t)), \tag{5.4}$$

where Δ is the innovation term that is chosen so as to drive the estimated outputs towards the true outputs, and $\widehat{\cdot}$ represents an estimation.

5.1.2 Object Pose Refinement via State Estimation

The proposed approach involves taking the output state from an existing object pose estimation network and refining this output using a state estimation framework to obtain more accurate pose estimates. I choose PVNet [15] to be the baseline object pose estimation network. In established algorithms such as PVNet, the state is estimated directly via a deep neural network. A typical formulation for such a problem can be written

$$\widehat{\mathbf{X}} = \mathcal{E}(\mathbf{V}), \tag{5.5}$$

$$\widehat{\mathbf{y}} = h(\widehat{\mathbf{X}}), \tag{5.6}$$

where the estimator $\mathcal{E}$ is implemented as a CNN or more classical computer vision algorithm.

The output of PVNet is a unit vector field, with vectors pointing from each pixel to each of a set of keypoints. The vectors are defined to be

$$\boldsymbol{\eta}_{ij}^{k} = \boldsymbol{\xi}^{k} - \boldsymbol{\xi}_{ij}, \tag{5.7}$$

where $\boldsymbol{\xi}_{ij}$ denotes a 2D pixel with coordinates (i, j) within an image $\mathcal{I}$ with dimensions $M \times N$, and $\boldsymbol{\xi}^k$ represents the pixel location of keypoint $k \in \mathcal{K}$. The unit vector field is

$$\boldsymbol{X}_{ij}^k = \frac{\boldsymbol{\eta}_{ij}^k}{\|\boldsymbol{\eta}_{ij}^k\|_2} \in \mathbb{R}^{2\times\mathcal{K}\times M\times N}. \tag{5.8}$$

This unit vector field forms the state for the estimation problem.

I assume a static state model

$$f(\mathbf{X}(t), \mathbf{V}(t)) = 0. \tag{5.9}$$

Approximating the time derivative by $\dot{\mathbf{X}} = \delta t(\mathbf{X}(t) - \mathbf{X}(t-1))$, the corresponding state estimation algorithm has the form

$$\widehat{\mathbf{X}}(t+1) = \widehat{\mathbf{X}}(t) - \Delta(t), \tag{5.10}$$

$$\widehat{\mathbf{y}}(t+1) = h(\widehat{\mathbf{X}}(t)), \tag{5.11}$$

where δt is absorbed into the innovation Δ.

The output of the state estimation algorithm $\widehat{\mathbf{y}}$ is the object pose. This is a discrete system with time steps $t \in \{0, 1, ..., T\}$.

The function h maps a vector field representation of keypoints to object pose via a RANSAC and uncertainty-driven PnP framework (EPnP), as discussed in [15]. The task then becomes to learn an appropriate Δ, such that the error in the state estimation algorithm is iteratively driven towards zero.

Consider the standard L2-norm loss function

$$\Phi(t) = \frac{1}{2}\|\widehat{\mathbf{X}}_{ij}^k(t) - \mathbf{X}_{ij}^k\|_2. \tag{5.12}$$

The gradient of this function is

$$\begin{aligned}\nabla\Phi(t) &= \frac{1}{2}\nabla_{\widehat{\mathbf{x}}_{ij}^k(t)}\|\widehat{\mathbf{X}}_{ij}^k(t) - \mathbf{X}_{ij}^k\|_2 \\ &= \widehat{\mathbf{X}}_{ij}^k(t) - \mathbf{X}_{ij}^k,\end{aligned} \tag{5.13}$$

where $\mathbf{X}_{ij}^k$ is the ground truth vector field, and ∇_x denotes the gradient operator with respect to x. Set

$$\Delta(t) = \alpha(t)\nabla\Phi(t), \tag{5.14}$$

Algorithm 3 Iterative Optimisation with an Innovation CNN

Choose $0 < \alpha(t) < 1$
Choose $T > 0$ ▷ Maximum #iterations
Input: I, $\mathbf{P}^k$ ▷ Image, 3D object points
$\widehat{\boldsymbol{X}}_{ij}^k(0) \leftarrow$ PVNet(I)
for $t = 1 \rightarrow T$ **do**
    $\widehat{\nabla\Phi}(t) \leftarrow \text{Innov}(I, \widehat{\boldsymbol{X}}_{ij}^k(t))$
    $\widehat{\boldsymbol{X}}_{ij}^k(t+1) = \widehat{\boldsymbol{X}}_{ij}^k(t) - \alpha(t)\widehat{\nabla\Phi}(t)$
$\widehat{\mathbf{y}}$ = EPnP($\widehat{\boldsymbol{X}}_{ij}^k(T)$, $\mathbf{P}^k$)
Output: $\widehat{\mathbf{y}}$ ▷ Pose

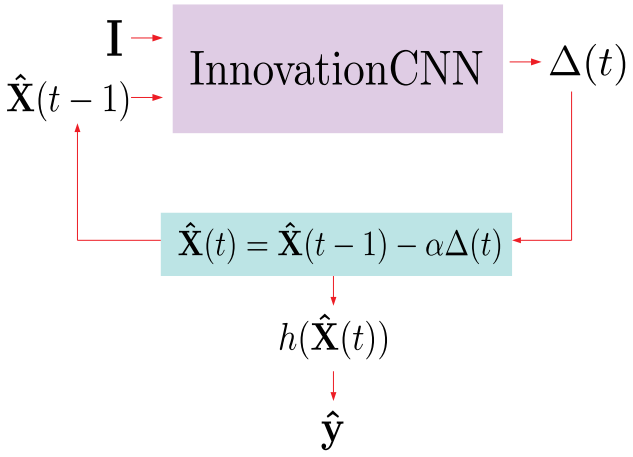


Figure 5.1: Visualise representation of iterative optimisation with an Innovation CNN.

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

In practice, the gradient term $\nabla\Phi(t)$ is not directly measured. Instead, I use an estimate of the value

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

where $\mathcal{F}$ represents the output of the CNN described in Section 5.1.3.

The iterative update is applied by employing gradient descent

$$\widehat{\boldsymbol{X}}_{ij}^k(t+1) = \widehat{\boldsymbol{X}}_{ij}^k(t) - \alpha(t)\widehat{\nabla\Phi}(t). \tag{5.15}$$

Within this iterative framework the initial state $\widehat{\mathbf{X}}_{ij}^k(0)$ is the vector field estimate obtained from PVNet, and $\widehat{\nabla\Phi}(t)$ is estimated by the Innovation CNN. Algorithm 3 and Figure 5.1 summarise the proposed information pipeline.

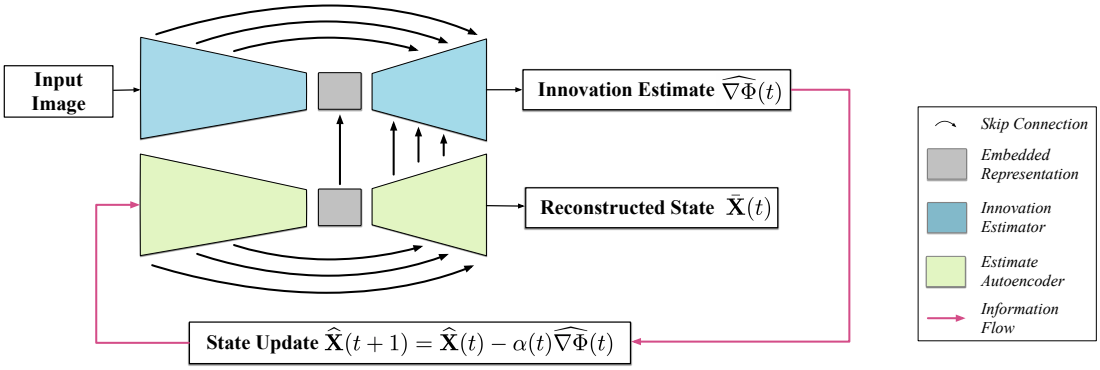


Figure 5.2: Network Architecture: Innovation Estimator (top block), and Estimate Autoencoder (bottom block). The input image (top left) is passed to the Innovation Estimator, which estimates $\widehat{\nabla\Phi}(t)$ (top right). The current state estimate $\widehat{\boldsymbol{X}}_{ij}^k(t)$ is passed to the State Encoder (bottom left). The output of the State Decoder (bottom right) is $\bar{\boldsymbol{X}}_{ij}^k(t)$, which represents the same information as $\widehat{\boldsymbol{X}}_{ij}^k(t)$. Using $\widehat{\nabla\Phi}(t)$ the state $\widehat{\boldsymbol{X}}_{ij}^k$ is updated via gradient descent (Eq. (5.15)). Both encoders reduce input resolution to the same intermediate dimensionality (the 'embedded representation'), after which skip connections are used to pass encoded state information to the Innovation Estimator.

5.1.3 Network Architecture

The Innovation CNN framework is developed to be applicable to a range of applications which are typically approached with an encoder-decoder network architecture. We begin by taking two copies of the baseline encoder-decoder network, in this case PVNet. We label the first network the *Innovation Estimator*, and modify the loss function to Eq. (5.18). We label the second network the *Estimate Autoencoder*, and modify the loss function to Eq. (5.16). We adapt the input later of the Estimate Autoencoder such that this network can receive a state estimate. We connect the two networks via skip connections as shown in Figure 5.2 to achieve a twin network architecture. The Innovation Estimator receives an image and returns an estimate of the gradient of the state $\widehat{\nabla\Phi}$. The Estimate Autoencoder receives a state $\widehat{\mathbf{X}}$ and returns an estimate of the state $\bar{\mathbf{X}}$. The Estimate Autoencoder is an auxiliary task aimed at allowing information from the initial state estimate to be injected into the Innovation Estimator via skip connections.

5.1.4 Training Loss

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

$$\mathcal{L}_{\text{state}} = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} \|\widehat{\boldsymbol{X}}_{ij}^{k}(t) - \bar{\boldsymbol{X}}_{ij}^{k}(t)\|_1, \tag{5.16}$$

where $(i,j) \in \mathcal{S}$ indicates that the pixel is within the segmentation mask, and the norm that is used is the 'smooth l1 norm' for unit vectors proposed by [126]. The state $\bar{\boldsymbol{X}}_{ij}^{k}$ represents the same information as $\widehat{\boldsymbol{X}}_{ij}^{k}$ and is used only for the purpose of training the autoencoder.

The state gradient loss is defined by

$$\mathcal{L}_{\text{innov}} = \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} \|\widehat{\nabla\Phi}(t) - \nabla\Phi(t)\|_1 \tag{5.17}$$

$$= \sum_{k=1}^{\mathcal{K}} \sum_{(i,j)\in\mathcal{S}} \|\widehat{\nabla\Phi}(t) - (\widehat{\boldsymbol{X}}_{ij}^{k}(t) - \boldsymbol{X}_{ij}^{k})\|_1. \tag{5.18}$$

The loss is backpropogated after each iteration of the gradient descent defined by Eq. (5.15).

The loss function used to train the network is

$$\mathcal{L} = \mathcal{L}_{\text{innov}} + \gamma\mathcal{L}_{\text{state}}, \tag{5.19}$$

where γ is a scaling factor.

5.1.5 Implementation Details

The network is trained for $T = 2$ iterations of Eq. (5.15) at each epoch, with a constant step size of $\alpha = 0.6$. I use a pretrained PVNet [15] to provide the initial estimate $\boldsymbol{X}_{ij}^{k}(0)$ for each object. A batch size of 64 is used, training images are down-sized by four to fit the learning algorithm on my machine. 8 keypoints are computed via farthest point sampling, from which object pose is obtained via uncertainty-driven PnP [15]. 10,000 images and synthesis 10,000 images are rendered for each object. Data augmentation is emploted in the form of random cropping, resizing, rotation, colour jittering. An Adam optimizer [127] is employed with an initial learning rate of $1e^{-3}$, which is decayed to $1e^{-5}$ by a factor of 0.85 every 10 epochs. The learning hyperparameter in Eq. (6.6) is set to $\gamma = 10$. The network is trained for 50 epochs.

During testing, Hough voting is employed to localise keypoints, before using

uncertainty-driven PnP to obtain object pose from keypoints. The step size is set to $\alpha = 0.01$ and perform iterations of Eq. (5.13) until the estimate converges. Once the gradient output of the Innovation CNN approaches zero the estimate is considered to be converged. Figures 5.4 and 5.3 provide qualitative results that illustrate this iterative convergence.

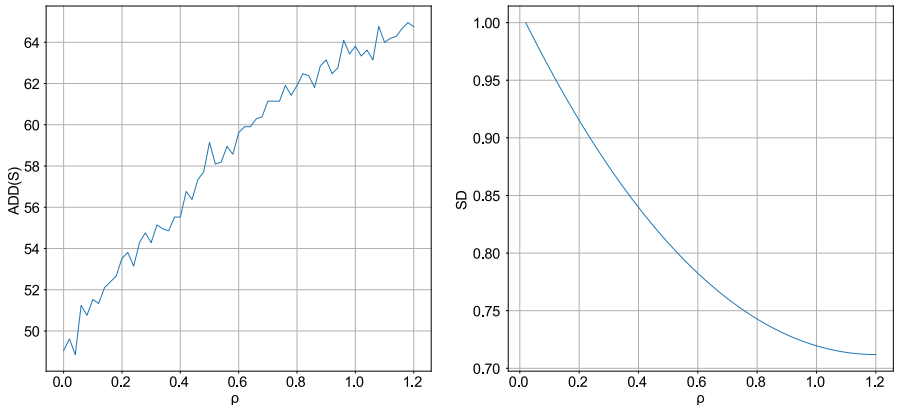


Figure 5.3: Iterative improvement on the **ADD(-S)** metric and corresponding percentage decrease in the State Distance (**SD**) for the ape object of the **LINEMOD** dataset plotted against the interpolation distance ρ (Eq. (5.24)) for $\alpha = 0.01$, $T = 120$.

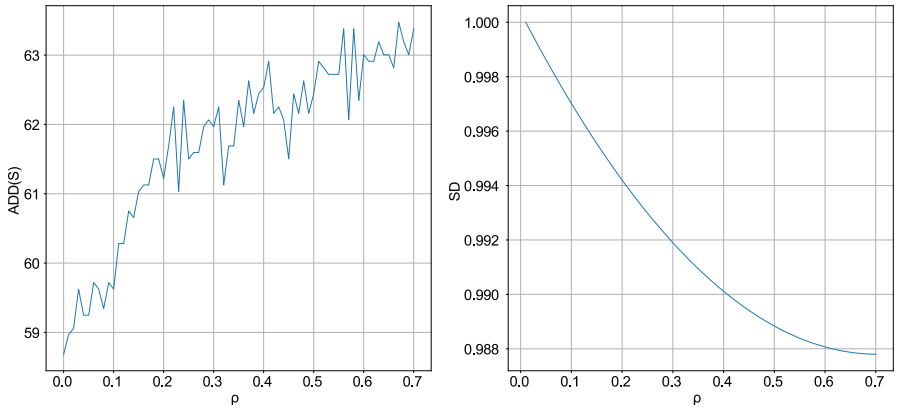


Figure 5.4: Iterative improvement on the **ADD(-S)** metric and corresponding percentage decrease in the State Distance (**SD**) for the duck object of the **LINEMOD** dataset plotted against the interpolation distance ρ (Eq. (5.24)) for $\alpha = 0.01$, $T = 70$.

5.2 Experiments

In this section experiments are conducted on two widely used datasets for object pose estimation, and evaluate the performance with two standard metrics for this task.

5.2.1 Datasets

LINEMOD [16] consists of 15783 images of 13 objects, with approximately 1200 instances for each object. Challenging aspects of this dataset include lighting variations, scene clutter, object occlusions, and texture-less objects.

Occlusion LINEMOD [17] is a subset of LINEMOD images with each image containing multiple annotated objects under severe occlusion, thus presenting a more challenging scenario for accurate pose estimation.

5.2.2 Evaluation Metrics

The **ADD** metric [16] computes the average 3D distance between the points of the 3D model under transformation from the ground truth and estimated pose respectively. Specifically, given the ground truth rotation $\mathbf{R}$ and translation $\mathbf{T}$ and the estimated rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{T}}$, the ADD metric computes

$$\text{ADD} = \frac{1}{m} \sum_{\mathbf{x} \in \mathcal{M}} \|(\mathbf{R}\mathbf{x} + \mathbf{T}) - (\hat{\mathbf{R}}\mathbf{x} + \hat{\mathbf{T}})\|_2, \tag{5.20}$$

where $\mathcal{M}$ denotes the set of 3D model points and m is the number of points. For symmetric objects the similar **ADD(-S)** metric [77] is used, where the mean distance is computed based on the closest point distance,

$$\text{ADD(-S)} = \frac{1}{m} \sum_{\mathbf{x}_1 \in \mathcal{M}} \min_{\mathbf{x}_2 \in \mathcal{M}} \|(\mathbf{R}\mathbf{x}_1 + \mathbf{T}) - (\hat{\mathbf{R}}\mathbf{x}_2 + \hat{\mathbf{T}})\|_2. \tag{5.21}$$

Both metrics are denoted as ADD(-S) in this work. In each case a pose is considered correct if the average distance is less than 10% of the 3D model diameter. The reported metric is the percentage of poses that are considered correct.

The **2D projection** metric [16] computes the mean distance between pixel locations of the 2D projections of the 3D model model, when projection is performed

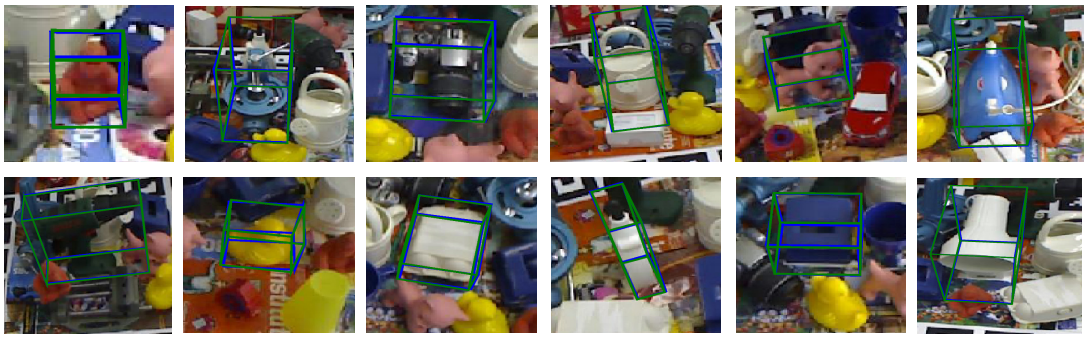


Figure 5.5: Visualisation of qualitative results on the **LINEMOD** dataset. Green bounding boxes represent ground truth poses and blue boxes represent the refined poses.

with the estimated and ground truth poses. The metric is defined via

$$\text{2d Proj} = \frac{1}{|V|} \sum_{\mathbf{v} \in V} \|\mathbf{P}\mathbf{Y}\mathbf{v} - \mathbf{P}\hat{\mathbf{Y}}\mathbf{v}\|_2, \tag{5.22}$$

where V is the set of all object model vertices, $\mathbf{Y}$ is the pose, and $\mathbf{P}$ is the camera matrix. The estimated pose is considered correct if the average error is less than 5 pixels. The metric reported is the percentage of correct poses.

Finally, I propose an additional metric, dubbed the **State Distance (SD)**, designed to quantify the difference between the estimated and ground truth vector field states. This is used to provide an indication of whether the state estimate $\widehat{\boldsymbol{X}}_{ij}^k$ is converging to or diverging from the true state $\boldsymbol{X}_{ij}^k$. This metric is defined by

$$\text{SD} = \frac{1}{s} \sum_{(i,j) \in \mathcal{S}} \|\widehat{\boldsymbol{X}}_{ij}^k(t) - \boldsymbol{X}_{ij}^k\|_2, \tag{5.23}$$

where $(i, j) \in \mathcal{S}$ indicates that the pixel is within the segmentation mask, and s denotes the number of pixels within the segmentation mask.

5.2.3 Comparison to State-of-the-Art

Results in terms of the ADD(-S) and 2D projection metrics on the LINEMOD and Occlusion LINEMOD datasets are presented in Tables 5.1, 5.2, and 5.3. Qualitative results illustrating the refined pose estimates are provided in Figures 5.5 and 5.6.

In this chapter I focus on refining an initial pose estimate without leveraging any extra supervision. In Tables 5.1,5.2,5.3 I compare my results to the top three other pose estimation networks in this category. The presented approach outperforms all

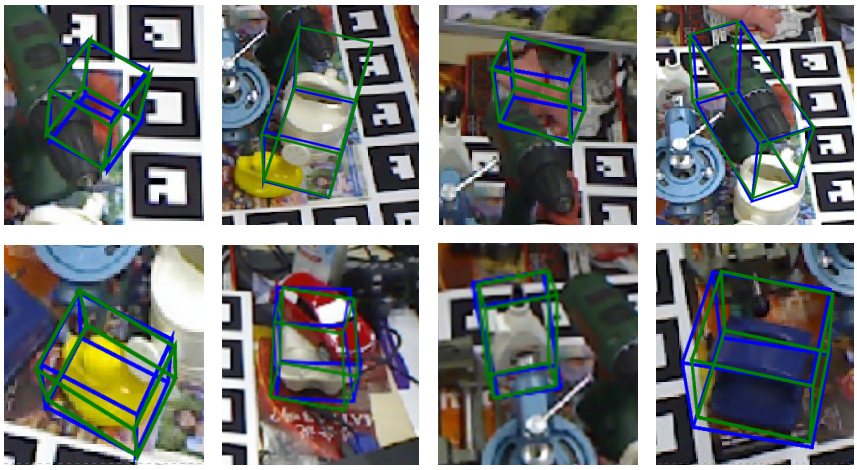


Figure 5.6: Visualisation of qualitative results on each object of the **Occlusion LINEMOD** dataset. Green bounding boxes represent ground truth poses and blue boxes represent the refined poses.

previous methods and achieves state-of-the-art performance on both the LINEMOD and Occlusion LINEMOD datasets, when compared to methods that do not require a textured 3D model.

Most significantly, the proposed network generates pose estimates that lead to a higher average on both the ADD(-S) and 2D projection metrics, compared to the baseline network. This includes a performance increase of 3.69% on the LINEMOD dataset and 3.31% on the Occlusion LINEMOD dataset in terms of the ADD(-S) metric. This is equivalent to improving the performance by 4.20% and 8.12% respectively compared to the performance of the baseline network.

It can be seen that the largest improvement is obtained for objects for which the baseline network provides a relatively poor initial estimate. Such objects include the relatively texture-less 'ape' and 'duck' and all objects in the Occlusion LINEMOD dataset. Specifically, on the ape and duck objects the baseline estimate is improved by 21.14% and 10.73% respectively in terms of the ADD(-S) metric. Qualitative results for these two objects are presented in Figure 5.4, for which the ADD(-S) metric is evaluated at each iteration of Eq. (5.15). Figure 5.4 also illustrates the iterative convergence of the state distance SD. In this figure, the ADD(S) and SD metrics are plotted as a function of the *interpolation distance*,

$$\rho = \alpha(t)T. \tag{5.24}$$

Methods	**DPOD** [11]	**CDPN** [128]	**PVNet** [15]	**OURS**
ape	53.28	64.38	43.62	**64.76**
benchwise	95.34	97.77	**99.90**	**99.90**
cam	90.36	91.67	86.86	**92.58**
can	94.10	95.87	95.47	**96.56**
cat	60.38	**83.83**	79.34	82.83
driller	**97.72**	96.23	96.43	97.52
duck	66.01	**66.76**	52.58	63.31
eggbox	**99.72**	99.72	99.15	99.32
glue	93.83	**99.61**	95.66	96.91
holepuncher	65.83	**85.82**	81.92	82.20
iron	**99.80**	97.85	98.88	99.31
lamp	88.11	97.86	99.33	**99.42**
phone	74.24	90.75	92.41	**94.33**
average	82.98	89.86	86.27	**89.92**

Table 5.1: Performance comparison on the **LINEMOD** dataset with respect to the **ADD(-S)** metric.

The interpolation distance quantifies the distance that has been iterated from the initial estimate. A illustration of the interpoluation distance is provided in Figure 5.8.

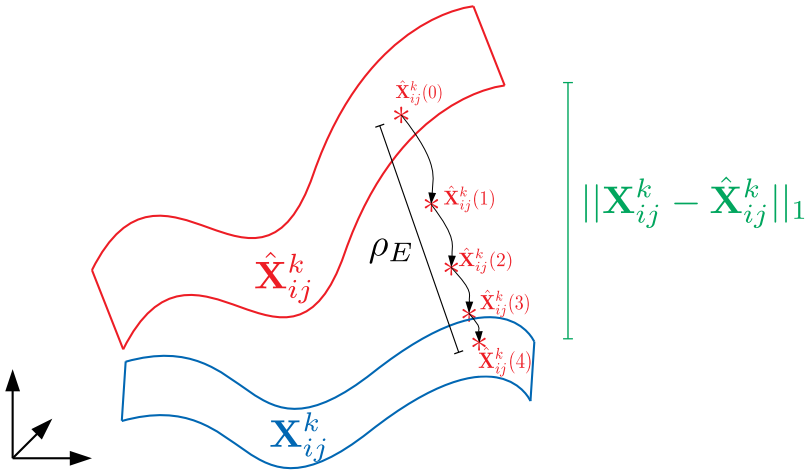


Figure 5.8: Gradient Descent applied to the state estimate $\hat{X}_{ij}^k$, leading to iterative convergence over the interpolation distance ρ.

Qualitative results illustrating the iterative application of the proposed method

Methods	**CDPN** [128]	**YOLO6D** [27]	**PVNet** [15]	**OURS**
ape	92.10	96.86	**99.23**	**99.23**
benchwise	95.06	98.35	**99.81**	**99.81**
cam	93.24	98.73	**99.21**	99.12
can	97.44	99.41	**99.90**	99.70
cat	97.41	99.80	99.30	**99.70**
driller	79.41	95.34	96.92	**97.23**
duck	94.65	**98.59**	98.02	98.02
eggbox	90.33	98.97	**99.34**	99.24
glue	96.53	**99.23**	98.45	98.45
holepuncher	92.86	99.71	**100.0**	99.70
iron	82.94	97.24	**99.18**	**99.18**
lamp	79.87	95.49	**98.27**	98.18
phone	86.07	97.64	**99.42**	**99.42**
average	90.37	98.10	**99.00**	**99.00**

Table 5.2: Performance comparison on the **LINEMOD** dataset with respect to the **2D Projection** error.

are presented for the Occlusion LINEMOD dataset in Figure 5.7. To generate Figure 5.7 the initial and final pose are displayed, along with two intermediate poses sampled from the iterative framework. For example, for the Ape object $T = 120$ iterations were used, and Figure 5.7 displays poses for T={0,40,80,120}.

5.2.4 Ablation Study

Experiments with a range of configurations were conducted before the configuration used to produce the presented results was chosen. Results of these experiments are summarised in Table 5.4. In Table 5.4, column one shows a comparison of training with (✓) and without (✗) the Estimate Autoencoder. Column two shows a comparison of backpropograting each iteration of Eq. (5.15) (✓) and backpropograting after T iterations (✗). Column three shows a comparison of choosing the output of PVNet as $\boldsymbol{X}_{ij}^{k}(0)$ (✓) and choosing the ground truth vector field computed from keypoints perturbed with 1% noise (✗). All ablation experiments were conducted on the Ape object of the **LINEMOD** dataset.

Results shown are the mean **ADD(-S)** results obtained from testing the final 20 epochs of the model. In Table 5.4 results of experiments with and without the Estimate Autoencoder (the bottom network block in Figure 5.2) are compared. The backpropgation scheme (backprop each iteration of Eq. (5.15)) is also compared with the alternative scheme of accumulating the loss and backpropogating after T iterations, as discussed in [40]. Finally a comparison is made between using a pretrained

Methods	**DPOD** [11]	**Pix2Pose** [129]	**PVNet** [15]	**OURS**
ape	-	22.0	15.81	**26.41**
can	-	44.7	**63.30**	61.31
cat	-	**22.7**	16.68	19.88
driller	-	44.7	65.65	**70.10**
duck	-	15.0	25.24	**31.99**
eggbox	-	25.2	**50.17**	49.44
glue	-	32.4	49.62	**51.16**
holepuncher	-	**49.5**	39.67	42.34
average	32.79	32.0	40.77	**44.08**

Table 5.3: Performance comparison on the **Occlusion LINEMOD** dataset with respect to the **ADD(-S)** metric.

	Estimate Autoencoder	Backprop each iteration	PVNet Initial Estimate
✓	**53.93**	52.01	**53.93**
✗	50.61	**53.93**	46.00

Table 5.4: Ablation Study of key design choices. The study indicates that the State Autoencoder improves performance, that it is better to backpropogate after each iteration of Eq. (5.15), and that it is better to use the better to use the output of PVNet as the initial estimate during evaluation.

PVNet as the initial estimate $\boldsymbol{X}_{ij}^{k}(0)$ and using the ground truth vector field or the ground truth vector field perturbed with different levels of noise. A range of different noise levels are trialed, and the result of applying 1% noise is presented in Table 5.4.

5.3 Conclusion

In this chapter I present a framework to improve an initial pose estimate. I present a novel CNN architecture, an Innovation CNN, capable of estimating the gradient from an initial state prediction to the true state. The Innovation CNN refines initial state estimates in an SGD framework iteratively, thus greatly reducing the difficulty of estimating object poses in a single forward pass. Extensive experiments on widely used datasets demonstrate that initial object pose estimates are improved by 4.20% on the LINEMOD dataset and 8.12% on the Occlusion LINEMOD dataset compared to baseline performance, in terms of the ADD(-S) metric. I also demonstrate that this approach is particularly effective when the initial pose estimate is relatively poor.

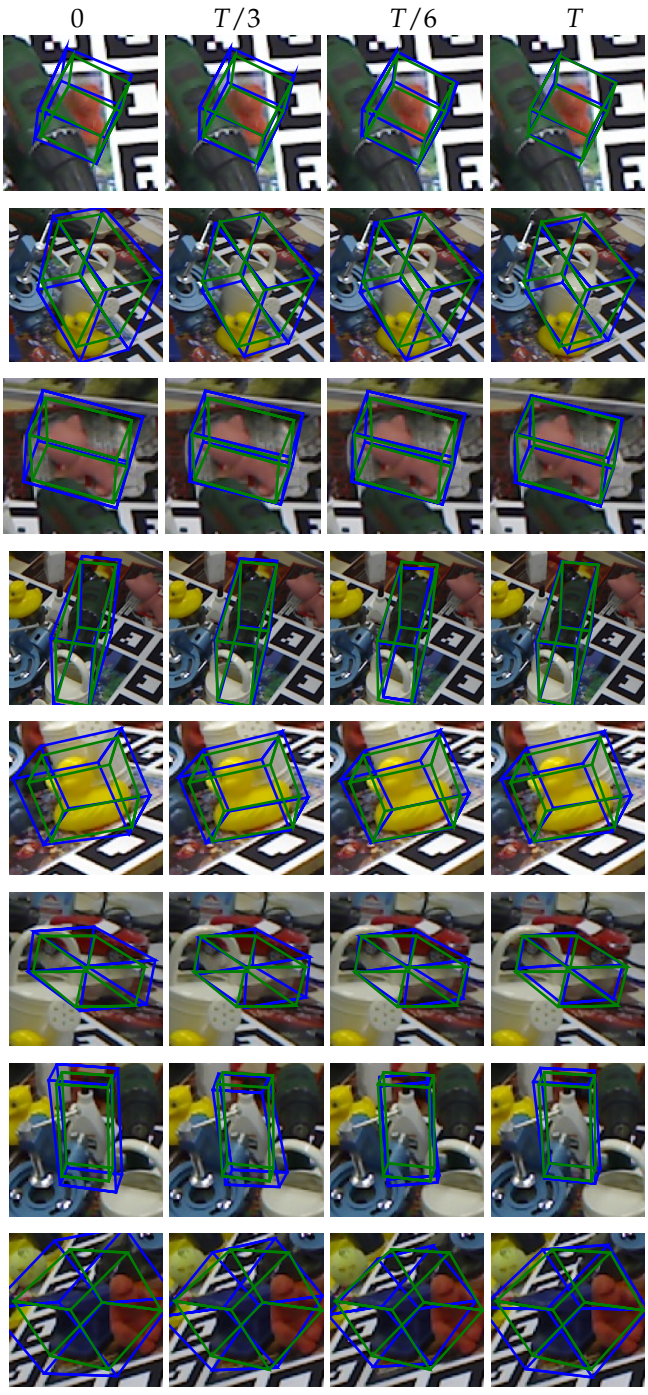


Figure 5.7: Iterative pose refinement on all objects in the **Occlusion LINEMOD** dataset. The initial pose estimate is shown in the left ($t = 0$), and the final pose estimate is shown on the right ($t = T$), with equidistant intermediate poses shown in-between. Green bounding boxes represent ground truth poses and blue boxes represent the refined poses.

Chapter 6

Iterative Optimisation with an Innovation CNN for Depth Refinement

Depth estimation from a single RGB is a key research problem within the fields of robotics and computer vision. Many algorithms within these areas, such as simultaneous localisation and mapping (SLAM), visual odometry, and 3D reconstruction often require an accurate per-pixel depth map. Traditionally depth information has been obtained via extra sensor modalities such as LIDAR or stereo cameras. These pose challenges including power and weight requirements, and operational constraints such as lighting conditions. For these reasons, depth estimation from a single image, computed with a deep learning algorithm, has become popular in recent years. In this chapter the generalisability of the Innovation CNN is demonstrated by applying the formulation developed in the previous chapter to the problem of depth refinement. Specifically, an Innovation CNN is trained on the popular NYU Depth V2 dataset [19] and applying the resultant state estimate in a stochastic gradient descent framework to refine an initial depth estimate. The initial depth estimate is obtained from FastDepth [18]. Results indicate an improvement of 4.5-4.9% relative to the initial estimate.

6.1 Proposed Innovation CNN

In this section I present a formulation for a general state estimation algorithm and apply this formulation to the problem of depth estimation. The network architecture, training loss, and network implementation are also outlined.

6.1.1 Depth Refinement via State Estimation

The proposed approach involves taking the output state from an existing depth estimation network and refining this output using a state estimation framework to obtain more accurate pose estimates. FastDepth [18] is chosen to be the baseline depth estimation network. The output of FastDepth is a per-pixel depth map, with values ranging from $0 \rightarrow \infty$.

Similar to the object pose estimation problem, the depth map is defined as

$$\boldsymbol{X}_{ij} \in \mathbb{R}^{M \times N}, \tag{6.1}$$

where i, j represent pixel coordinates of an image with dimensions $M \times N$. In the case of depth estimation the state comprises of one value (depth) per pixel. This depth map forms the state for the estimation problem.

A static state model is assumed, with the corresponding state estimation algorithm having the form

$$\widehat{\mathbf{X}}(t+1) = \widehat{\mathbf{X}}(t) - \Delta(t), \tag{6.2}$$

$$\widehat{\mathbf{y}}(t+1) = h(\widehat{\mathbf{X}}(t)). \tag{6.3}$$

The output of the state estimation algorithm $\widehat{\mathbf{y}}$ is the refined depth map. Unlike object pose estimation, for which the function h maps a vector field to an object pose, in the case of depth estimation the output is already in the required form. Thus the function h is an identity mapping. The task is again to learn an appropriate Δ.

The standard L2-norm loss function as defined in Eq. (5.12) is used, where the state $\widehat{\mathbf{X}}^k_{ij}(t)$ is replaced with $\widehat{\mathbf{X}}_{ij}(t)$. Likewise the gradient of this function is defined by Eq. (5.13) with a similar replacement. The iterative update is applied by employing gradient descent via Eq. (5.15). Within this iterative framework the initial state $\widehat{\mathbf{X}}_{ij}(0)$ is the depth estimate obtained from FastDepth, and $\widehat{\nabla\Phi}(t)$ is estimated by our network.

6.1.2 Network Architecture

The network follows the same high-level structure as shown in Figure 5.2 involving an Innovation Autoencoder network and an Estimate Autoencoder network, connected via skip connections. For the case of depth estimation the Innovation Encoder consists of a pre-trained MobileNet [130] architecture, while the Innovation Decoder consists of consecutive upsampling layers and nearest-neighbour interpolation. The

Estimate Autoencoder consists of the same architecture as the Innovation Autoencoder. The input dimensionality of the Estimate Autoencoder is modified to accept a depth map state estimate.

The Innovation Estimator receives an image and returns an estimate of the gradient of the depth field state $\widehat{\nabla\Phi}$. The Estimate Autoencoder receives a vector field state $\widehat{\boldsymbol{X}}_{ij}$ and returns an estimate of the depth map state $\bar{\boldsymbol{X}}_{ij}$.

6.1.3 Training Loss

Following the approach outlined in Section 5.1.4, the state loss is applied to the Estimate Autoencoder and defined by

$$\mathcal{L}_{\text{state}} = \frac{1}{N} \sum_{(i,j)} \| \widehat{\boldsymbol{X}}_{ij}(t) - \bar{\boldsymbol{X}}_{ij}(t) \|_1, \tag{6.4}$$

where N is the number of pixels in the image.

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

$$\mathcal{L}_{\text{innov}} = \frac{1}{N} \sum_{(i,j)} \| \widehat{\nabla\Phi}(t) - (\widehat{\boldsymbol{X}}_{ij}(t) - \boldsymbol{X}_{ij}) \|_1. \tag{6.5}$$

The loss function used to train the network is

$$\mathcal{L} = \mathcal{L}_{\text{innov}} + \gamma \mathcal{L}_{\text{state}}, \tag{6.6}$$

where γ is a scaling factor.

6.1.4 Implementation Details

The network is trained for $T = 2$ iterations of Eq. (5.15) at each epoch, with a constant step size of $\alpha = 0.6$. A pretrained FastDepth network [18] is used to provide the initial estimate $\boldsymbol{X}_{ij}(0)$. The network is trained with a batchsize of 8. A Stochastic Gradient Descent with Momentum (SGDM) optimizer [131] is employed with a learning rate of 0.04, which is reduced by 10% every 5 epochs. A mask is applied to filter out missing ground truth values in the dataset. The network is trained for 100 epochs.

6.2 Experiments

In this section experiments are conducted on a widely used dataset and performance is evaluated with two standard metrics for this task.

Model	RMSE ↓	RMSE Improvement (%) ↑	MAE ↓	MAE Improvement (%) ↑
Baseline	0.5868	N/A	0.4338	N/A
Innovation CNN	0.5599	4.58	0.4127	4.86

Table 6.1: Performance of the Innovation CNN compared to the Baseline model.

6.2.1 Dataset

Experiments are conducted on the **NYU Depth V2** dataset [19]. This dataset comprises of video sequences from a variety of indoor scenes, with a total of 1449 densely labeled pairs of aligned RGB and depth images.

6.2.2 Evaluation Metrics

The approach is evaluated using two standard metrics for depth estimation.

Root Mean Squared Error (RMSE) is the most commonly used metric in for evaluating depth estimation performance. RMSE computes the square root of the average of squared errors. Specifically, given a ground truth depth map $\mathbf{X}_{ij}$ and a predicted depth map $\widehat{\mathbf{X}}_{ij}$, RMSE computes

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i,j}^{N}(\mathbf{X}_{ij} - \widehat{\mathbf{X}_{ij}})}, \tag{6.7}$$

where N is the total number of pixels in the image.

Mean Absolute Error (MAE) computes the average of the absolute errors. Specifically, given a ground truth depth map $\mathbf{X}_{ij}$ and a predicted depth map $\widehat{\mathbf{X}_{ij}}$, MAE computes

$$\text{MAE} = \frac{1}{N}\sum_{i,j}^{N}|\mathbf{X}_{ij} - \widehat{\mathbf{X}}_{ij}|. \tag{6.8}$$

6.2.3 Comparison to Baseline Performance

Results in terms of the RMSE and MAE metrics on the NYU Depth V2 dataset are presented in Table 6.1. Figure 6.1 illustrates the iterative improvement obtained by applying the Innovation CNN in an SGD framework. Qualitative results illustrating our final depth estimates are provided in Figures 6.2 and 6.3. From these figures it can be seen that the Innovation CNN improves performance mainly around object edges and where there are gaps in the initial depth map.

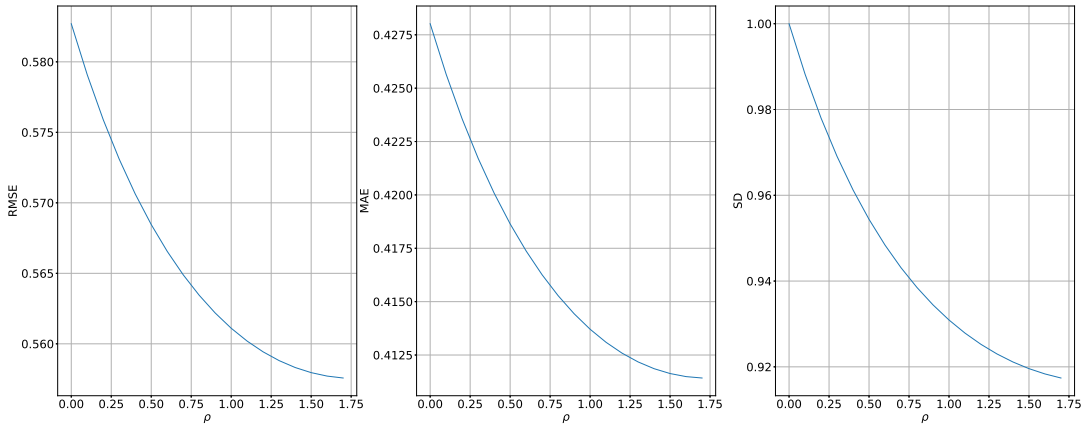


Figure 6.1: Iterative improvement on the **RMSE** and **MAE** metrics and corresponding percentage decrease in the State Distance (**SD**) on the **NYU Depth V2** dataset plotted against the interpolation distance ρ (Eq. (5.24)) for $\alpha = 0.1$, $T = 18$.

6.3 Conclusion

In this chapter I present a framework to improve an initial depth estimate. The framework is applied to estimate the gradient of the depth-map state, given an initial depth map provided by FastDepth [18]. Experiments on the widely used NYU Depth V2 dataset demonstrate that this approach improves initial depth estimates by 4.58% and 4.86% in terms of the RMSE and AME metrics respectively.

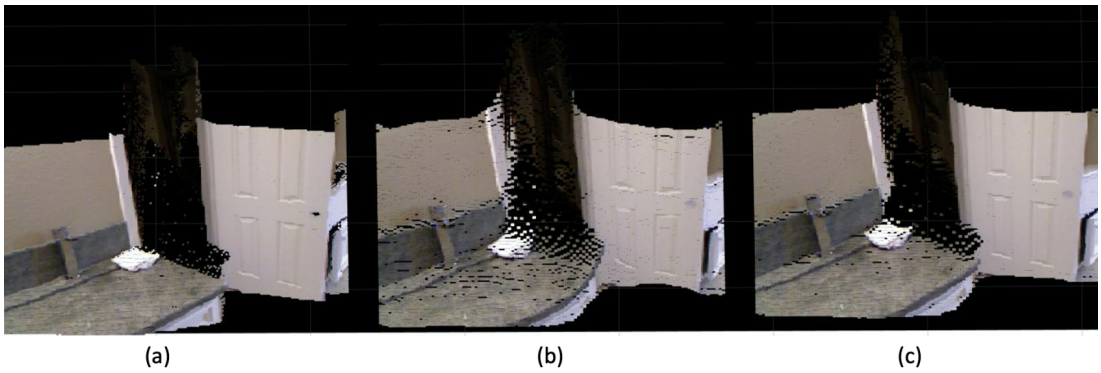


Figure 6.2: Example results compared to the ground truth (a) of the baseline model (b) and the Innovation CNN (c) [3].

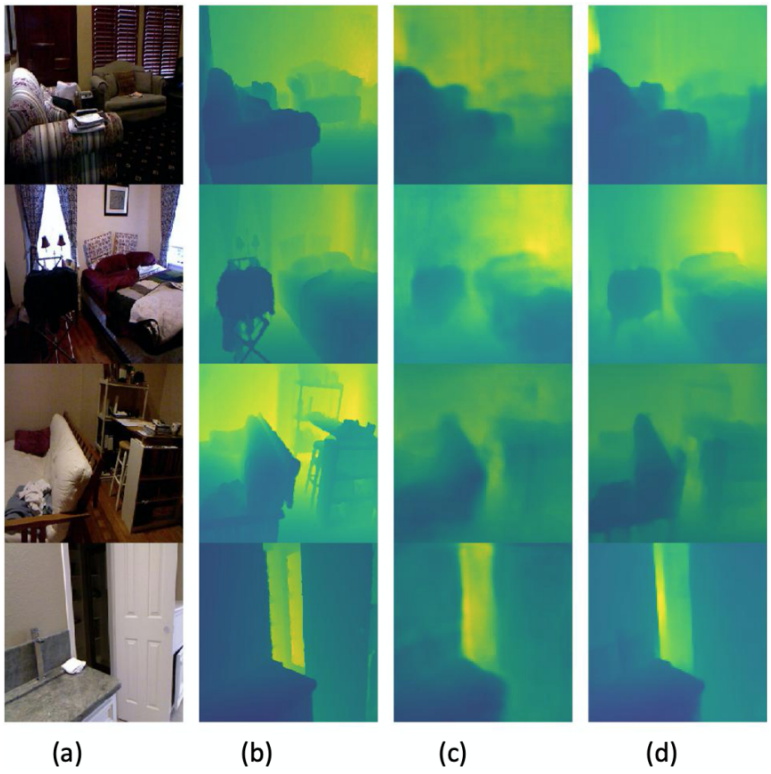


Figure 6.3: Visualisation of depth maps on the NYU Depth V2 dataset. (a) input image, (b) ground truth, (c) Baseline model, (d) Innovation CNN [3].

Chapter 7

Future Directions

This thesis aimed to develop solutions to perception-based robotic harvesting problems, for employment on real-world robotic platforms. While the project initially included aims to develop a prototype harvesting platform, this was ultimately deemed infeasible. Potential future work of this project therefore includes deployment of the presented ideas on robotic harvesters in the field. To achieve this goal it would be necessary to first either develop a prototype harvester, or partner with an organisation that would produce a prototype. To develop a prototype capable of operation in variable on-farm conditions significant funding and engineering work is required. To employ these ideas on a prototype harvester, a multi-camera system must be incorporated, along with significant compute resources. This will enable processing of image information at the rate necessary for the harvester to operate at speeds comparable to existing harvesting practices. I believe that reaching speeds comparable to existing methods is crucial for adoption of robotic harvesting technology, as it will allow for easy integration of the platform into existing farm operations.

Separate to the agricultural aims of this thesis, the Innovation CNN has potential applications in the wider robotics and computer vision communities. In this work the Innovation CNN was applied to two offline state estimation problems; object pose estimation and depth estimation. However, given the formulation of the Innovation CNN it is also well suited to dynamic, online applications which are common in robotics. The Innovation CNN takes an image and initial state and estimates the gradient to the true state. This approach is well suited to dynamic applications in which the input image is constantly updated.

Example problems could still include object pose estimation or depth estimation, however the approach would need to be applied to a dataset involving a sequence of images. Examples include the YCB Video dataset [77] for object pose estimation, or the KITTI dataset [105] for depth estimation. Applications of this approach include robotic manipulation with an eye-in-hand camera, in which the pose of the target

object would be updated each frame, and autonomous driving, where the depth map would be updated continuously as the car moved through the scene. Applying the Innovation CNN to dynamic, online state estimation problems would allow for well-studied, knowledge-based filtering techniques to be applied alongside a data-driven neural network. It is hoped that such an approach would draw on the best of both worlds, allowing image data to be interpreted with a neural network, then updating the desired representation via filtering.

Chapter 8

Conclusion

Worldwide, current methods for crop harvesting still rely heavily on manual labour. Due to increases in labour costs and global population numbers, there has been an increased focus on bring greater levels of automation to agriculture in recent decades. Current automation technology typically targets crops that can be harvested indiscriminately at a given time. There has been comparatively less progress on automation solutions that enable selective crop harvesting. Selective harvesting robotics is a key enabling technology that is expected to lead to more flexible growing practices, such as growing multiple crops in the same bed. A key roadblock to the progress of selective harvesting systems is the need to accurately identify and locate appropriate instances at acceptable speeds.

In this thesis I apply state estimation concepts to develop solutions to common problems related to perception systems in robotic harvesting platforms. These problems are: detecting, localising, and tracking green asparagus spears, object pose estimation from a single image, and depth estimation.

In the initial problem, I show that results comparable to dense 3D reconstruction can be obtained via simple algorithms that utlise the simple structure and lack of foliage that is unique to the green asparagus crop. The average difference between the proposed approach and a dense 3D reconstruction was found to be 0.74mm. Along with an approach to state estimation for robotic harvesting, I summarise the state of the market that the robotic harvesting platform would be integrated into, if it were to be commercialised. I also present a design for a prototype robotic platform for harvesting green asparagus.

Object pose estimation is a crucial task in robotic harvesting applications. It also presents unique challenges, as each crop instance is different, and a high-accuracy estimate is required to reduce rates of harvesting damage. In this thesis I present a framework for object pose refinement, based on deep learning and state estimation concepts. I apply this framework to existing datasets for object pose estimation, so

as to compare to existing methods. The approach improves performance by 4.20% on the LINEMOD dataset and 8.12% on the Occlusion LINEMOD dataset compared to baseline performance, in terms of the ADD(-S) metric.

Finally, the same framework was applied to the problem of depth estimation. Depth estimation is also a commonly used technique in robotic harvesting applications, as it can be used to obtain crop height and distance. Results show an improvement of 4.58% and 4.86% in terms of the RMSE and AME metrics respectively.

References

1. B. Holland, "Manipulation design for a robotic asparagus harvester," Undergraduate thesis, 2018.
2. S. S. Smith, "How to grow asparagus," https://garden.lovetoknow.com/wiki/How_to_Grow_Asparagus, 2021.
3. J. Gao, "Improving monocular rgb depth estimation with iterative innovation prediction," Undergraduate thesis, 2020.
4. FAO, "Global agriculture towards 2050," http://www.fao.org/fileadmin/templates/wsfs/docs/Issues_papers/HLEF2050_Global_Agriculture.pdf, 2009, accessed: 2019-05-23.
5. ACRV, "A robotics roadmap for australia," https://www.roboticvision.org/wp-content/uploads/Robotics-Roadmap_FULL-DOCUMENT.pdf, 2018.
6. T. Duckett, S. Pearson, S. Blackmore, B. Grieve, and M. Smith, "White paper - agricultural robotics: The future of robotic agriculture," 2018. [Online]. Available: https://uwe-repository.worktribe.com/output/866226
7. J. Billingsley, A. Visala, and M. Dunn, *Robotics in Agriculture and Forestry*, 01 2008, pp. 1065–1077.
8. G. Kootstra, X. Wang, P. Blok, J. Hemming, and E. Van Henten, "Selective harvesting robotics: Current research, trends, and future directions," *Current Robotics Reports*, 01 2021.
9. Cerescon, "Cerescon harvesting innovation," https://www.cerescon.com/EN/home, 2021.
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," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1530–1538, 2017.
11. S. Zakharov, I. S. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1941–1950, 2019.

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

13. F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6d pose refinement in rgb," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

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

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

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

17. E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6d object pose estimation using 3d object coordinates," in *ECCV*. Springer, September 2014. [Online]. Available: https://www.microsoft.com/en-us/research/publication/learning-6d-object-pose-estimation-using-3d-object-coordinates/

18. D. Wofk, F. Ma, T. Yang, S. Karaman, and V. Sze, "Fastdepth: Fast monocular depth estimation on embedded systems," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6101–6108.

19. P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

20. D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*, 01 2006.

21. T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

22. S. Thrun, "Probabilistic robotics," *Knowl. Eng. Rev.*, vol. 21, no. 3, p. 287–289, Sep. 2006. [Online]. Available: https://doi.org/10.1017/S0269888906210993

23. A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational Intelligence and Neuroscience*, vol. 2018, pp. 1–13, 02 2018.

24. Z. Zhao, P. Zheng, S. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

25. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

26. K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

27. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

28. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Lecture Notes in Computer Science*, p. 21–37, 2016. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46448-0_2

29. D. Yoo, S. Park, J. Lee, A. S. Paek, and I. S. Kweon, "Attentionnet: Aggregating weak directions for accurate object detection," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2659–2667.

30. Z. Fu, Y. Zheng, H. Ye, Y. Kong, J. Yang, and L. He, "Edge-aware deep image deblurring," *CoRR*, vol. abs/1907.02282, 2019. [Online]. Available: http://arxiv.org/abs/1907.02282

31. Y. Wenming, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, "Deep learning for single image super-resolution: A brief review," *IEEE Transactions on Multimedia*, vol. PP, pp. 1–1, 05 2019.

32. J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 341–349. [Online]. Available: http://papers.nips.cc/paper/4686-image-denoising-and-inpainting-with-deep-neural-networks.pdf

33. R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

34. D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

35. C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15–22.

36. J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

37. S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2043–2050.

38. K. Konda and R. Memisevic, "Learning visual odometry with a convolutional network," vol. 1, 03 2015.

39. B. Horn and B. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 08 1981.

40. J. Hur and S. Roth, "Iterative residual refinement for joint optical flow and occlusion estimation," in *CVPR*, 2019.

41. Z. Teed and J. Deng, "RAFT: recurrent all-pairs field transforms for optical flow," in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part II*, ser. Lecture Notes in Computer Science, A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds., vol. 12347. Springer, 2020, pp. 402–419. [Online]. Available: https://doi.org/10.1007/978-3-030-58536-5_24

42. M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 229 – 241, 07 2001.

43. R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3607–3613.

44. R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, p. 1147–1163, Oct 2015. [Online]. Available: http://dx.doi.org/10.1109/TRO.2015.2463671

45. R. Mahony and T. Hamel, "A geometric nonlinear observer for simultaneous localisation and mapping," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2408–2415.

46. P. van Goor, R. E. Mahony, T. Hamel, and J. Trumpf, "An equivariant observer design for visual localisation and mapping," *CoRR*, vol. abs/1904.02452, 2019. [Online]. Available: http://arxiv.org/abs/1904.02452

47. R. Kang, J. Shi, X. Li, Y. Liu, and X. Liu, "DF-SLAM: A deep-learning enhanced visual SLAM system based on deep local features," *CoRR*, vol. abs/1901.07223, 2019. [Online]. Available: http://arxiv.org/abs/1901.07223

48. S. Milz, G. Arbeiter, C. Witt, B. Abdallah, and S. Yogamani, "Visual slam for automated driving: Exploring the applications of deep learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

49. H. Zhou, B. Ummenhofer, and T. Brox, "Deeptam: Deep tracking and mapping," in *European Conference on Computer Vision (ECCV)*, 2018. [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2018/ZUB18

50. G. B. C Schertz, "Basic considerations in mechanizing citrus harvesting," *Transactions of the ASAE*, pp. 343–346, 1968.

51. E. A. Parrish, J. , and A. K. Goksel, "Pictorial pattern recognition applied to fruit harvesting," *Transactions of the ASAE*, vol. 20, pp. 0822–0827, 09 1977.

52. A. Jiménez, R. Ceres, and J. Pons, "A survey of computer vision methods for locating fruit on trees," *Trans. ASAE*, vol. 43, 07 2000.

53. C. W. Bac, E. J. van Henten, J. Hemming, and Y. Edan, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21525

54. ——, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21525

55. Y. Xiong, Y. Ge, L. Grimstad, and P. J. From, "An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation,"

Journal of Field Robotics, vol. 37, no. 2, pp. 202–224, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21889

56. C. Lehnert, A. English, C. McCool, A. W. Tow, and T. Perez, "Autonomous sweet pepper harvesting for protected cropping systems," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 872–879, April 2017.

57. M. Doldi, "What can robotics offer to agriculture?" https://www.agrimachinesworld.com/2019/06/11/what-can-robotics-offer-to-agriculture/, 2019.

58. K. G. Fue, W. M. Porter, E. M. Barnes, and G. C. Rains, "An extensive review of mobile agricultural robotics for field operations: Focus on cotton harvesting," *AgriEngineering*, vol. 2, no. 1, pp. 150–174, 2020. [Online]. Available: https://www.mdpi.com/2624-7402/2/1/10

59. M. Peebles, J. Barnett, M. Duke, and S. H. Lim, "Robotic harvesting of asparagus using machine learning and time-of-flight imaging – overview of development and field trials," 08 2020, pp. 1361–1366.

60. Geiger-Lund, "Geiger-lund selective asparagus harvesters," http://www.asparagusharvester.com/Drawing-Model-H-24.html, 2019, accessed: 2019-05-23.

61. J. Baeten, K. Donné, S. Boedrij, W. Beckers, and E. Claesen, "Autonomous fruit picking machine: A robotic apple harvester," in *FSR*, 2007.

62. S. Birrell, J. Hughes, J. Y. Cai, and F. Iida, "A field-tested robotic harvesting system for iceberg lettuce," *Journal of Field Robotics*, vol. 37, no. 2, pp. 225–245, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21888

63. A. Leu, M. Razavi, L. Langstädtler, D. Ristić-Durrant, H. Raffel, C. Schenck, A. Gräser, and B. Kuhfuss, "Robotic green asparagus selective harvesting," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2401–2410, Dec 2017.

64. RoboVeg, "Roboveg automation for agriculture," https://www.roboveg.com/, 2021.

65. C. Lehnert, C. Mccool, and T. Perez, "In-field peduncle detection of sweet peppers for robotic harvesting: a comparative study," 09 2017.

66. H. Kang, H. Zhou, and C. Chen, "Visual perception and modeling for autonomous apple harvesting," *IEEE Access*, vol. 8, pp. 62 151–62 163, 2020.

67. G. Arndt, R. Rudziejewski, and V. A. Stewart, "On the future of automated selective asparagus harvesting technology," *Computers and Electronics in Agriculture - COMPUT ELECTRON AGRIC*, vol. 16, pp. 137–145, 01 1997.

68. K. Haws, "Haws harvester," http://hawsharvester.com/, 2019, accessed: 2019-05-23.

69. T. R. K. Carpenter, E. Glasgow and D. Smith, "Mantis robotic asparagus harvester," https://sites.google.com/site/mrsdproject201213teamc/home, 2019, accessed: 2019-05-23.

70. M. Peebles, S. Lim, M. Duke, and B. McGuinness, "Investigation of optimal network architecture for asparagus spear detection in robotic harvesting," *IFAC-PapersOnLine*, vol. 52, no. 30, pp. 283–287, 2019, 6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896319324541

71. M. Peebles, S. H. Lim, M. Duke, and C. Au, "Overview of sensor technologies used for 3d localization of asparagus spears for robotic harvesting," *Applied Mechanics and Materials*, vol. 884, pp. 77–85, 08 2018.

72. M. Min, M. Peebles, S. H. Lim, M. Duke, and C. Au, "Model building and simulation of the suspension system for a robotic asparagus harvester," *Applied Mechanics and Materials*, vol. 884, pp. 69–76, 08 2018.

73. M. Peebles, S. H. Lim, L. Streeter, M. Duke, and C. Au, "Ground plane segmentation of time-of-flight images for asparagus harvesting," 11 2018, pp. 1–6.

74. N. Irie, N. Taguchi, T. Horie, and T. Ishimatsu, "Asparagus harvesting robot coordinated with 3-d vision sensor," in *2009 IEEE International Conference on Industrial Technology*, 2009, pp. 1–6.

75. G. Kennedy, V. Ila, and R. Mahony, "A perception pipeline for robotic harvesting of green asparagus," in *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019*, vol. 52, no. 30, 2019, pp. 288 – 293, iFAC-PapersOnLine. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896319324553

76. A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2938–2946.

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

78. S. Mahendran, H. Ali, and R. Vidal, "3d pose regression using convolutional neural networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 494–495.

79. M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," 2019.

80. D. Groos, H. Ramampiaro, and E. Ihlen, "Efficientpose: Scalable single-person pose estimation," *Applied intelligence*, 2020.

81. B. Tekin, S. N. Sinha, and P. Fua, "Real-Time Seamless Single Shot 6D Object Pose Prediction," in *CVPR*, 2018.

82. M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3848–3856, 2017.

83. X. Yu, Z. Zhuang, P. Koniusz, and H. Li, "6dof object pose estimation via differentiable proxy voting regularizer," in *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*. BMVA Press, 2020. [Online]. Available: https://www.bmvc2020-conference.com/assets/papers/0287.pdf

84. C. Song, J. Song, and Q. Huang, "Hybridpose: 6d object pose estimation under hybrid representations," 2020.

85. A. Aristidou and J. Lasenby, "Inverse kinematics: a review of existing techniques and introduction of a new fast iterative solver," 2009.

86. J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

87. J. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. Sagastizabal, *Numerical Optimization – Theoretical and Practical Aspects*, 01 2006.

88. S. Phaniteja, P. Dewangan, P. Guhan, A. Sarkar, and K. M. Krishna, "A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots," in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2017, pp. 1818–1823.

89. O. Wiles and A. Zisserman, "Silnet : Single- and multi-view reconstruction by learning from silhouettes," *BMVC*, 2017.

90. D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, p. 2366–2374.

91. F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

92. Bo Li, Chunhua Shen, Yuchao Dai, A. van den Hengel, and Mingyi He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1119–1127.

93. D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," 11 2014.

94. R. Cipolla, Y. Gal, and A. Kendall, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7482–7491.

95. H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011.

96. J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," *arXiv preprint arXiv:1907.10326*, 2019.

97. S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," *ArXiv*, vol. abs/2011.14141, 2020.

98. R. Garg, B. V. Kumar, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European Conference on Computer Vision*. Springer, 2016, pp. 740–756.

99. C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, 2017.

100. T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, 2017.

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

102. M. Rossi, M. E. Gheche, A. Kuhn, and P. Frossard, "Joint graph-based depth refinement and normal estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

103. S. Gidaris and N. Komodakis, "Detect, replace, refine: Deep structured prediction for pixel wise labeling," 12 2016.

104. P. Knöbelreiter and T. Pock, *Learned Collaborative Stereo Refinement*, 10 2019, pp. 3–17.

105. A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Rob. Res.*, vol. 32, no. 11, p. 1231–1237, Sep. 2013. [Online]. Available: https://doi.org/10.1177/0278364913491297

106. M. Andrychowicz, M. Denil, S. G. Colmenarejo, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. de Freitas, "Learning to learn by gradient descent by gradient descent," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 3988–3996.

107. J. Adler and O. Öktem, "Solving ill-posed inverse problems using iterative deep neural networks," *Inverse Problems*, vol. 33, no. 12, p. 124007, Nov 2017. [Online]. Available: http://dx.doi.org/10.1088/1361-6420/aa9581

108. P. Putzky and M. Welling, "Recurrent inference machines for solving inverse problems," *CoRR*, vol. abs/1706.04008, 2017. [Online]. Available: http://arxiv.org/abs/1706.04008

109. H. et al, "Evaluation of automation and robotic innovations: developing next generation vegetable production systems," AusVeg, Project Number: VG13113, pp. 78, 2016.

110. T. . Werling, "Costs and returns for producing asparagus," Michigan State University Extension Bulletin E-33155, pp. 21, 2016.

111. H. A. Ltd, "Mechanical harvesting of selected vegetables – industry scoping study," Project VG05073, 2006.

112. T. Cembali, R. J. Folwell, C. D. Clary, and M. Mari, "Economic comparison of selective and nonselective mechanical harvesting of asparagus," *International Journal of Vegetable Science*, vol. 14, no. 1, pp. 4–22, 2008. [Online]. Available: https://doi.org/10.1080/19315260801890476

113. F. M. Insights, "Asparagus market: canned and frozen segments expended to account for substantial growth over the forecast period – global industry analysis and opportunity assessment 2017-2027," https://www.futuremarketinsights.com/reports/asparagus-market, 2017.

114. A. B. of Statistics, "Value of selected agricultural commodities produced, australia, preliminary, 2005-2006," https://www.abs.gov.au/AUSSTATS/abs\spacefactor\@m{}.nsf/DetailsPage/7502.02005-06?OpenDocument, 2007.

115. MarketsandMarkets, "Agricultural robots by market type, offering, application and geography – global forecast to 2022," https://www.prnewswire.com/news-releases/agricultural-robots-market-by-type-offering-application-and-geography---global-forecast-to-2 html, 2017.

116. ——, "Agriculture equipment market by tractor, drive type, combines, implements and region – global forecast to 2023," https://www.marketresearch.com/MarketsandMarkets-v3719/Agriculture-Equipment-c458/1.html, 2018.

117. J. Davidson, S. Bhusal, C. Mo, M. Karkee, and Q. Zhang, "Robotic manipulation for specialty crop harvesting: A review of manipulator and end-effector technologies," *Global Journal of Agricultural and Allied Sciences*, vol. 2, pp. 25–41, 09 2020.

118. KPMG, "Powering growth: Realising the potential of agtech for australia," https://home.kpmg/au/en/home/insights/2016/09/powering-growth-realising-potential-agtech-australia.html, 2016.

119. A. Walmsley, "Development needed to keep ag bots progressing," https://www.goodfruitandvegetables.com.au/story/5373272/development-needed-to-keep-ag-bots-progressing/, 2018.

120. Omron, "World's fastest and first 4 arm delta robot at ppma," https://industrial.omron.eu/en/news-events/news/, 2017.

121. P. Moulon, P. Monasse, R. Marlet, and Others, "Openmvg. an open multiple view geometry library." https://github.com/openMVG/openMVG, 2017.

122. Matlab, "Single camera calibration app," https://au.mathworks.com/help/vision/ug/single-camera-calibrator-app.html, 2019.

123. G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

124. Wolfram, "Steiner circumellipse," http://mathworld.wolfram.com/SteinerCircumellipse.html, 2019.

125. B. Anderson and J. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall, 1979.

126. R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

127. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

128. Z. li, G. Wang, and X. Ji, "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation," in *ICCV*, 10 2019.

129. K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7667–7676, 2019.

130. A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 04 2017.

131. I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1139–1147. [Online]. Available: http://proceedings.mlr.press/v28/sutskever13.html