# Robotic Crop Interaction in Agriculture for Soft Fruit Harvesting

**Jasper Brown BE (Hons 1)**

A thesis submitted in fulfillment
of the requirements of the degree of
Doctor of Philosophy

THE UNIVERSITY OF
SYDNEY

Australian Centre for Field Robotics
School of Aerospace, Mechanical and Mechatronic Engineering
The University of Sydney

Submitted March 2021; revised September 2021

# Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.

I am the primary and corresponding author for all the following papers, parts of which form contents of this thesis.

For Brown and Sukkarieh (2019) and Brown and Sukkarieh (2021) I was responsible for theoretical development, study design, implementation, data gathering, analysis and manuscript writing, with research guidance and coordination support from Salah Sukkarieh. The former of these papers forms the study in Section 3.4, while the latter covers field trial and results aspects of Chapter 6.

For Brown et al. (2019) and Brown et al. (2020) I was responsible for simulation and experiment work, results analysis and significant sections of manuscript writing. The co-authors contributed to the simulation experiment, theoretical development, writing and research coordination & funding. This work forms the core of Section 4.4.

**Jasper Brown**

20 September 2021

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

**Salah Sukkarieh**

20 September 2021

# Abstract

Jasper Brown, BE (Hons 1)
The University of Sydney

Doctor of Philosophy
September 2021

## Robotic Crop Interaction in Agriculture for Soft Fruit Harvesting

Autonomous tree crop harvesting has been a seemingly attainable, but elusive, robotics goal for the past several decades. Limiting grower reliance on uncertain seasonal labour is an economic driver of this, but the ability of robotic systems to treat each plant individually also has environmental benefits, such as reduced emissions and fertiliser use. Over the same time period, effective grasping & manipulation (G&M) solutions to warehouse product handling, and more general robotic interaction, have been demonstrated.

Despite research progress in general robotic interaction and harvesting of some specific crop types, a commercially successful robotic harvester has yet to be demonstrated. Most crop varieties, including soft-skinned fruit, have not yet been addressed. Soft fruit, such as plums, present problems for many of the techniques employed for their more robust relatives and require special focus when developing autonomous harvesters. Adapting existing robotics tools and techniques to new fruit types, including soft skinned varieties, is not well explored. This thesis aims to bridge that gap by examining the challenges of autonomous crop interaction for the harvesting of soft fruit.

Aspects which are known to be challenging include mixed obstacle planning with both hard and soft obstacles present, poor outdoor sensing conditions, and the lack of proven picking motion strategies. Positioning an actuator for harvesting requires solving these problems and others specific to soft skinned fruit. Doing so effectively means addressing these in the sensing, planning and actuation areas of a robotic system. Such areas are also highly interdependent for grasping and manipulation tasks, so solutions need to be developed at the system level.

In this thesis, soft robotics actuators, with simplifying assumptions about hard obstacle planes, are used to solve mixed obstacle planning. Persistent target tracking and filtering is used to overcome challenging object detection conditions, while multiple stages of object detection are applied to refine these initial position estimates. Several picking motions are developed and tested for plums, with varying degrees of effectiveness. These various techniques are integrated into a prototype system which is validated in lab testing and extensive field trials on a commercial plum crop.

Key contributions of this thesis include

I. The examination of grasping & manipulation tools, algorithms, techniques and challenges for harvesting soft skinned fruit

II. Design, development and field-trial evaluation of a harvester prototype to validate these concepts in practice, with specific design studies of the gripper type, object detector architecture and picking motion for this

III. Investigation of specific G&M module improvements including:

- Application of the autocovariance least squares (ALS) method to noise covariance matrix estimation for visual servoing tasks, where both simulated and real experiments demonstrated a 30% improvement in state estimation error using this technique.
- Theory and experimentation showing that a single range measurement is sufficient for disambiguating scene scale in monocular depth estimation for some datasets.
- Preliminary investigations of stochastic object completion and sampling for grasping, active perception for visual servoing based harvesting, and multi-stage fruit localisation from RGB-Depth data.

Several field trials were carried out with the plum harvesting prototype. Testing on an unmodified commercial plum crop, in all weather conditions, showed promising results with a harvest success rate of 42%. While a significant gap between prototype performance and commercial viability remains, the use of soft robotics with carefully chosen sensing and planning approaches allows for robust grasping & manipulation under challenging conditions, with both hard and soft obstacles.

# Acknowledgements

This thesis would never have been finished, or indeed started, without the many people in my life who have contributed to it directly through research guidance, or indirectly by keeping me sane.

I am grateful for the support of my supervisors Salah Sukkarieh and He Kong. Salah, for providing the opportunity to pursue this research at the ACFR, along with guidance and vision. He, for always being available, insightful and passionate. You are one of the hardest working people I have met.

Thanks to my family Kevin, Jenny & Sas, for always being interested in my research, even when I was not. Your compassion and support made both my PhD and undergraduate degree possible.

My friends in Ag research, Jen, Nathan, Su, Stu & Yongliang. The Rose St gang Felix, Fletcher, Jacob, Jackson, James, Johnny, Stein, Tara, Teja, Vera, Wei & Wilhelm. Plus the old guard of John, Lloyd, Nader, Phil & Steve. Max, who I've pestered more than anyone with occasionally astute, but mostly just dumb, questions. You have all made the long nights, paper rejections, experiment setbacks, and assignment marking weeks bearable. Being able to spend time among brilliant, funny and driven people is the best part of the job.

Eric and Tom for putting up with my hare-brained field trial schedules in the middle of nowhere, without missing a beat on the technical side. Sorry for the amount of pizza hood you had to eat. Also Khalid, for making that possible and suffering through it with us.

The ACFR tech staff Nedim and Mark, thanks for helping turn my ideas into reality. Yi Sun for his soft robotics expertise and components. Also the several Daves, Jag, Javier, Matt & Tony for technical advice and help. The office staff Annette, Lesley, Liz & Ruth.

Those who taught me the foundations of robotics and did it well enough that I would consider another 4 years at university. David Rye, Dries, Eduardo, Graham & Stef.

To all the other mentors, scientists, engineers and generally inquisitive people who have influenced my life. We so rarely get to thank the giants whose shoulders we stand on.

And to Júlia, I'm sorry I convinced you to do a PhD.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Nomenclature

## List of Symbols

**General**

| | |
|---|---|
| $A^T$ | Vector or matrix transpose of $A$ |
| $x_k$ | The value of $x$ at time step $k$ |
| $\dot{x}$ | Derivative of $x$ |
| $\hat{x}$ | Estimated value of $x$ |
| $f(\cdot)$ | Generic function |
| $\varrho, \tau$ | Generic function or model parameters |
| $I_n$ | An $n \times n$ identity matrix |
| $\mathbf{0}_{n_r,n_c}$ | An $n_r \times n_c$ matrix with all zero entries |
| $\mathbb{E}[\cdot]$ | Expected value of a random variable |
| $x \sim \mathcal{N}(\mu, \Sigma)$ | A normally distributed random vector with mean $\mu$ and covariance matrix $\Sigma$ |
| $\mathrm{Var}(\cdot)$ | Variance of a random variable |
| $\|A\|_F$ | Frobenius norm of a matrix $A$ |
| $\|A\|_2$ | L2 (Euclidean) norm of a vector $A$ |
| $\nabla$ | Gradient operator |
| $\frac{\partial f(\cdot)}{\partial x}$ | Function partial derivative with respect to $x$ |

**Camera Model & Reference Frames**

| | |
|---|---|
| x, y, z | Points in axes |
| $X^C, Y^C, Z^C$ | Camera cartesian frame |
| $X^O, Y^O, Z^O$ | Object cartesian frame |
| $u^C, v^C$ | Camera image plane frame |
| $P^C$ | Point in camera frame |
| $P^O$ | Point in object frame |
| $\boldsymbol{P^C}$ | Point in image plane frame |
| $\boldsymbol{C}$ | Camera intrinsics matrix |
| $\boldsymbol{R}_{OC}$ | Rotation matrix from object to camera frame |
| $\boldsymbol{T}_{OC}$ | Translation matrix from object to camera frame |

| | |
|---|---|
| $(qw, qx, qy, qz)$ | Quaternion values |
| $f_{x,y}$ | Camera focal length in X and Y axes |

**Monocular Depth Prediction**

| | |
|---|---|
| $d$ | Depth of an object |
| $h_{u,v}$ | Object dimensions in the image frame |
| $h_{cX,cY,cZ}$ | Canonical object dimensions in X,Y,Z |
| $h_{X,Y,Z}$ | Specific object dimensions in camera frame |

**Stochastic Shape Completion for Grasping**

| | |
|---|---|
| $V_{x,y,z}$ | Voxel at position (x, y, z) |
| $p_o$ | Probability of occupancy for a voxel |
| $\varphi$ | Model precision |

**Estimation & Filtering**

| | |
|---|---|
| $x_k$ | 6 degree of freedom state vector of camera pose |
| $\tilde{x}_k$ | 12 degree of freedom state vector of camera pose |
| $y_k$ | State vector of fruit position |
| $z_k$ | Observations vector |
| $\hat{z}_k$ | Filter predicted observations vector |
| $h(\cdot)$ | Observation function |
| $H_k$ | Jacobian of the observation function at the current state estimate |
| $\Sigma$ | Filter state vector covariance matrix |
| $Q$ | System noise covariance matrix |
| $V$ | Measurement noise covariance matrix |
| $w_k$ | Sample drawn from $\sim \mathcal{N}(0, Q)$ |
| $v_k$ | Sample drawn from $\sim \mathcal{N}(0, V)$ |
| $\bar{w}_k$ | Stacked combination of $v, w$ |
| $\epsilon$ | Filter state error |
| $\bar{\varphi_k}$ | Filter innovations |
| $\phi, \alpha, \psi$ | Roll, pitch, yaw |
| $\Theta_k^*$ | MHE optimal state estimate |

**Autocovariance Least Squares**

| | |
|---|---|
| $\otimes$ | The Kronecker product |
| $(\cdot)_s$ | Column wise vectorisation of a matrix |
| $(\cdot)_{ss}$ | The lower triangular elements of a symmetric matrix which are column-wise stacked |
| $\mathbb{R}$ | The set of real numbers |
| $\mathbf{0}_n$ | An $n$ length column vector with all zero entries |

| | |
|---|---|
| $\Sigma \succeq 0$ | Indicates $\Sigma$ is positive semidefinite ($\Sigma$ is also symmetric) |
| $\mathcal{I}_{n,q}$ | A permutation matrix of only ones and zeros such that $\Big(I_n \otimes R\Big)_s = \mathcal{I}_{n,q}(R)_s$ |
| $\mathscr{D}$ | A duplication matrix used to relate $(\cdot)_s$ and $(\cdot)_{ss}$ |
| $\mathscr{C}_j$ | Auto-covariance of a measurement with a $j$ time lag copy |
| $\mathcal{R}$ | Autocovariance matrix |
| $\bar{\mathcal{R}}$ | Autocovariance matrix estimated from data |
| $\widehat{\Sigma}, \widehat{Q}, \widehat{V}$ | ALS estimates for the NC matrices |
| $T$ | Time interval between filter steps |
| $N$ | ALS maximum time lags |
| $M$ | Total ALS data points |
| $M_e$ | Data window length used for ALS estimation |
| $L$ | Initial sub-optimal filter gain constructed using $\Sigma_g, Q_g, V_g$ |
| $n$ | Image frames downsampling rate |

**Active Perception**

| | |
|---|---|
| $W_\Sigma$ | Pose estimate covariance weighting matrix |
| $\rho_x^e(\Sigma)$ | EKF covariance update step |
| $\rho^p(\Sigma)$ | EKF covariance prediction step |
| $\text{trace}(\Sigma)$ | Matrix trace, the sum of diagonal elements |
| $\mathcal{I}$ | Information |
| $\mathbb{I}(\cdot,\cdot)$ | Mutual Information |
| $\mathcal{U}$ | Set of possible actions |
| $u_t$ | A specific action choice |
| $\vec{u}$ | A vector used to generate $\mathcal{U}$ from $x_0$ to $\hat{y_0}$ |
| $\sigma$ | A single camera trajectory |
| $\mathcal{C}$ | Modified camera intrinsics matrix |
| $f_{prj}$ | Projective camera linearisation function |

**Harvesting Motion Controllers**

| | |
|---|---|
| $u_{vel}, v_{vel}, d_{vel}$ | X,Y,Z relative IBVS errors in pixel space |
| $G_u, G_v$ | X and Y dimension IBVS gains in pixel space |
| $u_{Bbox}, v_{Bbox}$ | Current bounding box centroid coordinates |
| $u_{target}, v_{target}$ | Grip point coordinates in camera frame |
| $ee_{vel}$ | Preset end effector velocity |
| $\vec{V}_{cmd}$ | Commanded end effector velocity vector |

## List of Acronyms

| | |
|---|---|
| **ALS** | autocovariance least squares |
| **AP** | active perception |
| **APM** | average precision metric |
| **CAD** | computer aided design |
| **CNN** | convolutional neural network |
| **DNN** | deep neural network |
| **DoF** | degree of freedom |
| **EKF** | extended Kalman filter |
| **ER** | environmental representation |
| **FIE** | full information estimation |
| **FoV** | field of view |
| **FPN** | feature pyramid network |
| **FVI** | forward value iteration |
| **G&M** | grasping & manipulation |
| **GP** | Gaussian process |
| **GPIS** | Gaussian process implicit surface |
| **GPS** | global positioning system |
| **GQM** | grasp quality metric |
| **HSV** | hue-saturation-value |
| **IBVS** | image based visual servoing |
| **IK** | inverse kinematics |
| **IMU** | inertial measurement unit |
| **IOU** | intersection over union |
| **LSTM** | long short-term memory |
| **LTI** | linear time invariant |
| **LTV** | linear time variant |
| **MCMC** | Markov chain monte carlo |
| **MHE** | moving horizon estimation |
| **NC** | noise covariance |
| **NN** | neural network |
| **PBVS** | position based visual servoing |
| **PID** | proportional-integral-derivative *(the controller type)* |
| **PR** | precision-recall |
| **RGB** | red-green-blue *(refers to a standard 2D camera)* |
| **RGBD** | red-green-blue-depth |
| **RMS** | root mean squared |
| **ROS** | robot operating system *(the open source project)* |
| **RoI** | region of interest |
| **RVI** | reduced value iteration |
| **SDF** | signed distance field |
| **SDP** | semi-definite programming |

| | |
|---|---|
| **SLAM** | simultaneous localisation and mapping |
| **SOR** | stochastic object reconstruction |
| **SRT** | stochastic regularisation technique |
| **SVM** | support vector machine |
| **ToF** | time of flight |
| **TSDF** | truncated signed distance field |
| **UPS** | uninterruptible power supply |

# Chapter 1

# Introduction

Global food demand is forecast to increase by 15% over the coming decade and meeting this requirement while reducing environmental and climate impacts will require technological and policy innovation, see OECD and Food and Agriculture Organization of the United Nations (2019). Tree crop growers in Australia face these same issues, with the added complication of a large seasonal labour force which is unreliable and increasingly uneconomical to source, as documented by Martin et al. (2020).

Robotics provides significant opportunities for increasing productivity in horticulture while reducing production inputs, and many of the most valuable agricultural applications require physical crop interaction. Robotic grasping & manipulation (G&M) are difficult tasks which have been the focus of research for several decades and are still under active development. Applying these techniques in agriculture is even more challenging due to uncontrolled outdoor environments containing water, dust, and variable lighting. Non-rigid, fragile and highly complex objects such as plants or fruits make many traditional grasp planning tools ineffective and simulation of horticultural tasks very difficult.

Despite these challenges, many research and commercial groups are now working on solutions for harvesting specific indoor or hard skinned, crop types such as apples, sweet peppers, strawberries and cucumbers. Soft skinned tree crops are a valuable market which present unique and largely unexplored challenges because the tech-

niques, components and algorithms common in robotics literature have not been developed for these and require adaptation. For example, the vacuum harvesters applied to apples rely on their robust skin to avoid damage, while strawberry trellis systems are different to those of tree crops like plums. Understanding the process of autonomous soft skinned tree crop harvesting is the goal of this thesis. It explores the selection and adaptation of common robotics techniques to soft fruit harvesting, within the context of a plum picking prototype. Each stage of the grasping & manipulation pipeline is constructed with this goal in mind, then tested as part of a week-long field trial, which is the first of its kind to target the harvesting of an unmodified commercial plum crop.

Several stand-alone studies are conducted into specific theoretical and practical system improvements. These cover stochastic shape completion from partial observations of objects, monocular depth inference under ambiguous scale scenes, a thorough comparison of object detector architectures for eye-in-hand harvesting, improving filter noise parameter estimates in visual servoing applications using the autocovariance least squares method, and active perception techniques for bearings only fruit localisation.

Key grasping & manipulation stages, and how each thesis chapter considers these, are shown in Figure 1.1. The environmental representation (ER) is how sensor data is stored in a coherent view of the world. Sensor processing is used to translate direct input frames into fruit properties such as size and position which are stored in the ER. This includes detection, pose estimation, and filtering. Grasp planning selects a grasp motion and position, while control elements operate at multiple system scales to regulate actuator and gripper motion. Manipulation follows the grasping step, for harvesting, this includes the detachment motion and placement in a bin.

The thesis concludes with a field trial evaluation of the harvesting prototype platform on a commercial plum crop. This platform consists of an articulated robot arm and custom designed gripper, with two eye-on-hand cameras, which is fixed to a self contained mobile trailer base. A localisation camera tracks the trailer position in the orchard, which also has a basket for depositing harvested fruit into during the trial.

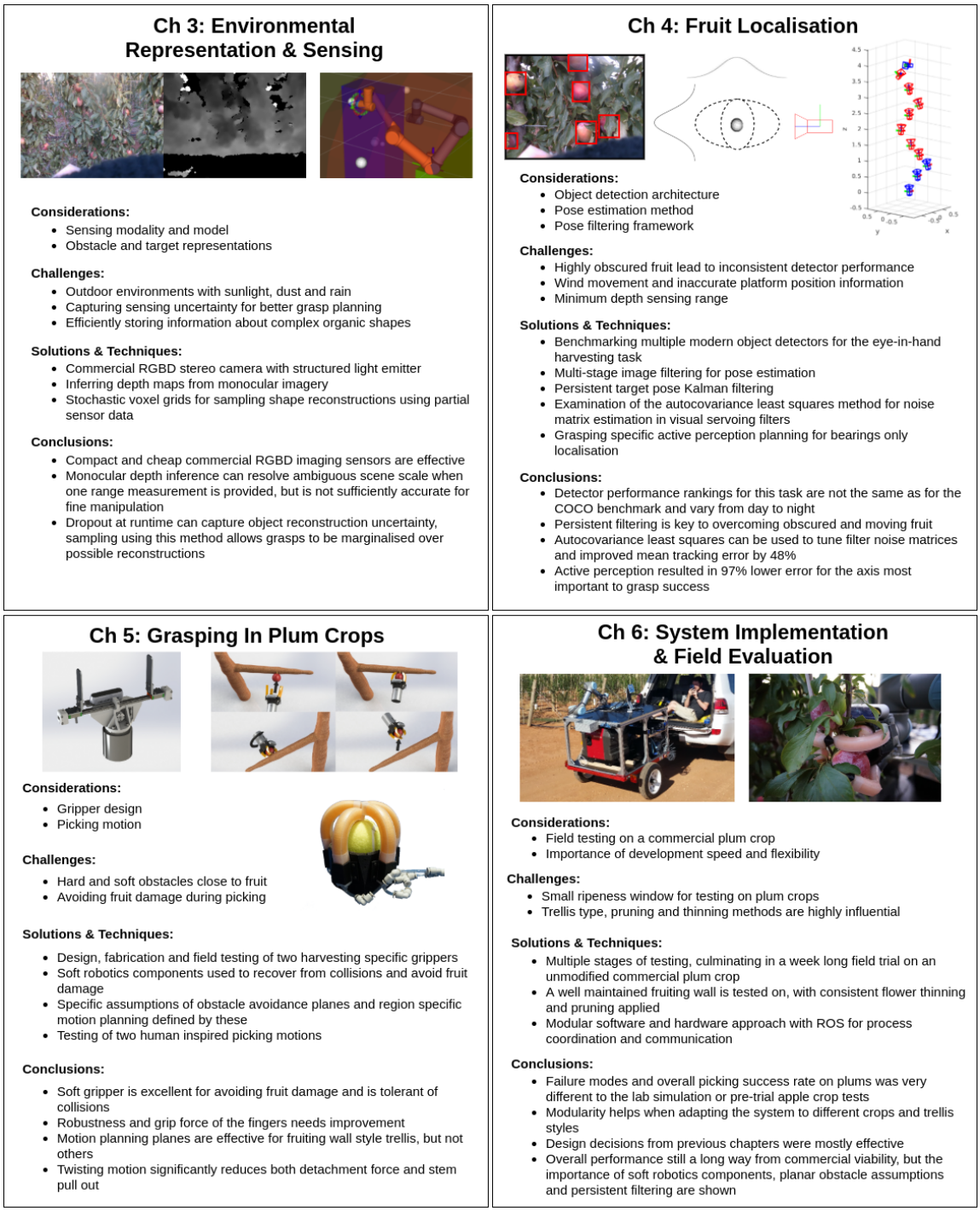


**Figure 1.1** – Overview of thesis chapters and the key grasping & manipulation points in each.

## 1.1 Thesis Objectives

Autonomous fruit harvesting requires a long and complex pipeline of functionality to be successful. Within existing solutions, the focus tends to lie on improving individual components. Despite the success of specific algorithms and designs in research contexts, deployment of grasping and manipulation systems in real world agriculture tasks is very limited. Many of these system components come from the wider robotics literature and are not well tested or optimised for fruit harvesting. Additionally to the individual component performance, the interactions between these to magnify or cancel out picking failures are not well understood.

This thesis will examine these interactions to better understand the process of autonomous harvesting of soft fruit, in order to improve the real world performance of grasping and manipulation in agriculture. The design decisions when constructing a prototype plum harvester will be assessed and improvements to key modules in the G&M pipeline will be made.

The specific objectives of this thesis are to

- Critically assess the design decisions made with regard to the entire grasping and manipulation pipeline including both the performance of individual components and the relationships between these at a system level, in order to best support soft fruit harvesting
- Present experimental results for the prototype robotic plum harvesting system and each component of this to allow for future progress in harvesting less common crop types
- Develop and test improvements to specific key components to improve the performance of robotics technology within harvesting and crop interaction
- Provide an illustrative example of a modern grasping and manipulation pipeline within robotic agriculture, using current algorithms and techniques from the perception, planning and grasping robotics literature

## 1.2 Contributions

The main contribution of this thesis is an investigation of techniques and tools for autonomous soft fruit harvesting, with an experimental validation of these on a novel plum picking prototype. Both rigid and soft gripper designs are explored for plum harvesting, along with simple and complex types of picking motion. Lessons around soft robotics components, mixed obstacle planning, target filtering, active perception for harvesting and picking motion are identified, which may also be applied to other less common tree crop types.

Additionally, several stand-alone studies of theoretical and practical improvements to key grasping and manipulation pipeline components are conducted, including:

- Improvements to extended Kalman filter (EKF) and moving horizon estimation (MHE) filter tuning for visual servoing tasks. This allows more accurate filters to be used for fruit localisation.
- A new dataset, benchmarked models and improvements to object detection for camera-in-hand plum harvesting tasks. This allows for better plum detection under all environmental conditions.
- A new approach to predicting depth maps from monocular imagery. This makes the use of this technique possible for ambiguous scenes, which are frequently present in agriculture.
- A new approach to sampling stochastic object representations for grasping when predicting obscured parts of objects. This allows for scene completion with limited sensor data and performs the highest quality grasp given uncertain environment parameters.
- Additional investigations of active perception for fruit localisation using bearings only camera measurements

## 1.3 Thesis Structure

**Chapter 2** provides a brief background of historical and modern robotic grasping theory along with reviews of current literature on each of the system components. Existing approaches to autonomous fruit harvesting are examined.

**Chapter 3** covers environmental representations and sensor selection for harvesting. Stochastic shape completion for grasping is studied, along with one method for improving depth estimation from monocular imagery in ambiguous scenes by fusing a single range measurement.

**Chapter 4** focuses on the fruit localisation process which interprets raw sensor data and incorporates this in the environmental representation. This includes steps of object detection, pose estimation and filtering. Object detection was found to be problematic, so several modern deep learning models are benchmarked against a new harvesting oriented object detection dataset. Specific improvements to estimating the filter noise covariance matrices, using the autocovariance least squares (ALS) approach for position based visual servoing (PBVS) are presented. A reduced value iteration active perception framework is applied to determine the efficacy of bearings-only fruit localisation.

**Chapter 5** covers issues specific to grasp execution in plum crops. The problem of harvesting in the presence of mixed hard and soft obstacles is addressed here. Various methods, factors and solutions to this are examined, including; the design of both a hard and soft gripper, the selection and control of an actuation system to position these grippers and the choice of picking motion. Two final approach controllers are also presented.

**Chapter 6** describes the overall architecture and implementation of the developed plum harvesting prototype system, along with key trellis parameters. Field trial experiment design and results are included, with overall performance figures and a failure mode analysis. Finally, **Chapter 7** is the conclusion and future work directions.

# Chapter 2

# Background & Literature Review

In this chapter, the concepts of grasping and manipulation are defined, and both historical and modern research in these areas is briefly summarised. Publications targeting autonomous fruit harvesting are examined, followed by literature review sections specific to the detailed component studies presented in this thesis.

## 2.1 A Background on Grasping and Manipulation

Grasping and manipulation have been desired functionality in robotics for several decades, although applications of these outside repetitive factory settings remain elusive. Likewise, the concept of autonomous farming has been common long before the technology to realise it was developed. These two areas intersect in the problem of autonomous crop interaction, including such tasks as tree crop flower thinning, pruning and general weeding. Robotic harvesting is the interaction task that this thesis focuses upon.

The term *grasping* will be used in reference to the process of placing an end effector in a configuration which allows an object to be held in a static pose relative to that end effector. This is typically accomplished by physically closing end effector components, such as fingers, around the object, but can also apply to vacuum, magnetic or other

gripper types. *Manipulation* is physical interaction to alter or utilise an object for a specific purpose.

An illustrative example common in robotics competitions is turning a valve. This requires planning a grasp, positioning for the planned grasp by placing the robot hand over the valve handle, then closing the robot hand to grasp the handle. These three steps are grasping, with the added intent to rotate the valve. Turning the valve is manipulation. Most manipulation actions require a grasping step, though approaches such as pushing may not.

In the context of fruit harvesting, placing the gripper in contact with a fruit is grasping. While removing that fruit from the stem and placing it in a collector is manipulation. These two phases are closely coupled, but have distinct requirements, so both terms will generally be used together.

Grasping theory spans a wide range of approaches, algorithms and hardware. One cohesive framework for thinking about these disparate areas is by considering grasping as a problem of state space parameterisation and search. Grasp planning is typically the core problem to be solved, being more difficult than grasp execution. This consists of determining values for an end effector pose parameterisation, that are optimal in some sense, given uncertainties about the environment. The relevant search space is the end effector configuration, which may be parameterised by every end-effector joint, only fully actuated joints, or a subset of these. It is also common to plan in low-dimensional parameter spaces, such as the 6 degree of freedom (DoF) pose of the palm or gripper centroid, and then execute an open loop gripper closing action.

Given a selected parameterisation the configuration space must be searched for a maximum value of the optimality criteria. One basic, but widely used, technique is to perform a sampling based search while executing a physics simulation at each configuration to predict the resulting grasp stability. A recently demonstrated paradigm is to parameterise grasps directly in sensor data space, such as over a top-down depth map, and densely search this reduced space.

Which optimality criteria to use is another rich area of research, common ones include

maximising the probability of success, minimising the probability of grasp failure under uncertainties, or maximising informativeness for future grasps. Environment uncertainty often covers object properties such as geometry, friction coefficients, mass, or pose. It also extends to robot state and obstacle uncertainties.

## 2.2 Historical and Modern Grasping Literature

Early work in grasping theory by Mason and Salisbury (1985) considers the 2D case and describes the complimentary conditions of form and force closure, illustrated in Figure 2.1. Both of these build on the concept of a wrench vector, which is the force and moment able to be applied by a given contact between a hand and object. With the goal of immobilising an object using applied wrenches, form closure is when contacts are modelled as frictionless points. This is equivalent to the configuration point of the object being surrounded by c-obstacles in configuration space, as used in Rimon and Burdick (1996). Force closure allows frictional properties for the applied wrenches, and occurs when the convex hull of wrenches in the wrench space includes the origin.

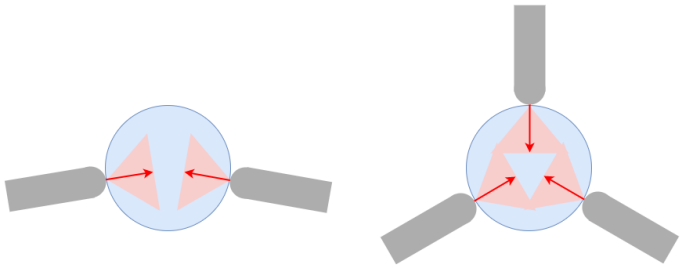

**Figure 2.1** – An example of force (left) and form (right) closure grasps. Top down view of a blue cylinder moving only in the XY plane is shown, with normal forces depicted using red arrows and 2D friction cones using red shading. The form closure grasp can immobilise the object position in the absence of friction.

These two conditions of immobility are closely tied to grasp quality metrics. Ferrari and Canny (1992) propose their eponymous measure of the minimum distance from the origin to the convex hull of the wrench space for a given grasp configuration. This

is the minimum wrench required to dislodge an object from a grasp. Other quality metrics are assessed by Roa and Suárez (2015).

Matching the task goal to a suitable disturbance wrench space is considered in Borst et al. (2004), who observed that non-uniformity between the force and torque dimensions of a wrench limits the application of quality metrics. For unknown tasks, the disturbance wrench space should still be made robust to gravity and object acceleration wrenches due to arm movement.

Many animals, humans included, are highly adept at grasping, so numerous works have sought to apply lessons from nature to grasping theory. Napier (1956) catalogues a series of prehensile movements for human hands when manipulating objects. Cutkosky and Howe (1990) extend this to a formal taxonomy of human grasps, which are divided into power and precision types. By considering analytical and knowledge-based approaches together, they are able to draw insights into why certain grasps are more appropriate for some objects or tasks. Practical approaches to grasp planning, as reviewed in Bicchi and Kumar (2000), rarely search over the full parameterisation space of a complex gripper. Some parameterise the grasp by the 6 DoF pose of the palm, then automatically close the joints to achieve a grasp. Ciocarlie et al. (2007) use human generated pregrasp poses with principal component analysis to define a low-dimensional basis for planning dexterous hand grasps, then search for good configurations within this eigengrasp space. Searching for grasps in reduced parameter space may be relevant to fruit harvesting, where repetitive motions and consistent target shapes are likely to result in repetitive grasps. This can reduce both actuator degree of freedom requirements, and planning time. As mentioned below, fruit picking motions and possible subspaces of these, remain open research questions.

Recent works have effectively applied machine learning tools to densely predict reward maps over top-down view depth images, such as the reactive approach of Morrison et al. (2018). Online reinforcement learning with large amounts of data is tested by Levine et al. (2018) to learn grasping policies from scratch. Kleeberger et al. (2020) provide a review of this field. Both online and offline learning of motion policies is very data intensive, which makes this technique difficult to employ in harvesting.

Recovery from obstacle collisions or trellis entanglement frequently requires human intervention, limiting the deployment of online learning systems without expensive human supervision.

A large amount of grasping literature is omitted from this section for brevity, including such techniques as adaptive grippers, tactile feedback, multimodal sensing for clear or reflective objects, human controlled teleoperation, learning in-hand manipulation and Bayesian grasp planning.

## 2.3 Robotic Tree Crop Harvesting

Robotic harvesting of tree crops has a long history with published works going back over 30 years and over 50 papers on the topic, many being recent works, such as Bac et al. (2014); Comba et al. (2010); De-An et al. (2011). Davidson et al. (2020) provides an recent review of this area. The diverse and challenging conditions encountered in agriculture mean that even this significant research effort covers only a small number of crop types, and to a low rate of success. Commercialisation aspects such as system speed, maintenance requirements, and cost of manufacture are often unaddressed by the research literature.

Protected growing conditions offer lighting regularity, higher value crops and tightly controlled growing structures making these attractive initial targets for autonomous harvesting, as in van Henten et al. (2013). Arad et al. (2020) present a sweet pepper platform, and Xiong et al. (2019) one for strawberries. Cucumbers and cherries are the focus of van Henten et al. (2002) and Tanigaki et al. (2008) respectively. Common to all of these crops are soft and flexible stems which result in collisions causing damage to plants, but not interrupting or damaging the robotic system. Outdoor tree crops in contrast, have hard lignified branches which easily harm harvester components.

One of the first demonstrations of an outdoor tree crop harvester is seen in Baeten et al. (2008) which targets apples using a suction gripper and eye-in-hand sensing. A more recent apple harvester is presented by Silwal et al. (2017) using a low cost

3D printed gripper and a custom built prismatic base manipulator arm executing rotational picking motion. They report system performance on a modified crop, making realistic picking success rate assessment difficult. Positioning error is one problem highlighted in this work, along with longer apple stems acting as pendulums during picking.

Moving from hard to soft fruit types introduces additional complexity which is not well understood at present. Defining a line between hard and soft fruit is challenging, and large variations also occur between plum cultivars, as shown in Esehagh Beygi et al. (2014), but plums typically have lower skin toughness than crops such as apples and pears. The soft skin is easily damaged by contact with hard tree wood or trellis components during picking, while stem pull out occurs at lower forces. Both conditions make the fruit unsaleable due to bacterial ingress through the broken skin.

Plums are chosen as a representative and informative soft fruit type for both intrinsic and extrinsic reasons. They are an informative choice because little work has been done on automated or mechanised harvesting of these, and the lower skin toughness introduces complexities not seen in fruit such as pears or citrus. Extrinsically, plums were available to us with the correct fruiting wall trellis type during the planned field trial timeline of this work, with a supportive grower providing access for field experiments. System integration tests did take place on apple crops, due to ease of access.

Mechanised plum harvesting is examined in Mika et al. (2015), who reported that up to 18% of harvested plums were damaged and an additional small amount were dropped or missed. This rate of damage is acceptable where cold storage is employed and fruit are destined for further processing into jams and or baking products. Table fruit require more delicate handling to prevent damaging the soft skin on hard tree wood or trellis components, requirements that cannot be fulfilled by mechanised bulk harvesters.

Soft robotics techniques are one approach to dealing with positioning errors and collisions. This field considers the use of compliant or biologically inspired sensors, actuators, and embodiments, as described by Verl et al. (2015). The apple harvester

of Silwal et al. (2017) is further developed in Hohimer et al. (2019) including the addition of a soft pneumatic gripper and further testing on unmodified crops. Two learned feature detectors are used in an ensemble for fruit detection, with multiple camera exposures taken to increase dynamic range under outdoor lighting. Picking motion is also examined, with off-horizontal pitch angles found to increase collisions without improving picking success. This study determined fruit clustering to be the most prevalent failure case, with positioning error also remaining a problem.

Numerous commercial entities are bringing product prototypes to market at the time of writing, although none have exited the development phase. Target crop types include strawberries (Agrobot[1], Octinion[2], CROO Robotics[3]), raspberries (fieldwork robotics[4]), tomatoes (Panasonic[5]) and apples (Abundant Robotics[6], FFRobotics[7]). The narrow focus of these to specific crop types and growing systems means autonomous harvesting of lower volume fruit varieties will be an open research problem for some time.

Individual fruit harvesting requires robust object detection. Research has focused on both hand-engineered detection features, and learned features, as in Kapach et al. (2012); Nguyen et al. (2014); Sa et al. (2017); Vitzrabin and Edan (2016) and Bargoti and Underwood (2017a). Object detection for eye-in-hand imagery gathered during harvesting attempts remains an unexplored topic. Further review of this area is presented in Section 2.4.3.

Gripper design is both critical to grasping performance and task specific. Along with the pneumatic gripper of Hohimer et al. (2019), an under-actuated cable driven hand is shown in Xiong et al. (2019). Tactile feedback can be used to inform grasp success as in Dimeas et al. (2015). Certain fruit have additional requirements, such as cutting the stem of sweet peppers, as demonstrated by Bac et al. (2017) and Lehnert et al.

[1]agrobot.com. Accessed on 27/4/2020
[2]octinion.com/products/agricultural-robotics/rubion. Accessed on 28/4/2020
[3]harvestcroo.com. Accessed on 28/4/2020
[4]phys.org/news/2019-05-fieldwork-robotics-field-trials-raspberry.html. Accessed on 28/4/2020
[5]news.panasonic.com/global/stories/2018/57801.html. Accessed on 28/4/2020
[6]abundantrobotics.com. Accessed on 28/4/2020
[7]ffrobotics.com. Accessed on 28/4/2020

(2017). No existing literature reports the performance of rigid or soft gripper designs for a commercial plum crop, so effective hardware designs for these remain to be determined.

Final target approach is done using image based visual servoing (IBVS) in Barth et al. (2016); Mehta et al. (2016) and Arad et al. (2020). A review of vision based harvesting control can be found in Zhao et al. (2016). Infrared distance sensing is applied for motion feedback by Xiong et al. (2019). Multiple approach angle strategies are tested by Ringdahl et al. (2019), who found additional attempts did increase sweet pepper harvest success rate at the cost of longer execution time. A study of orchard fruit reachability is carried out by Vougioukas et al. (2016) with over 90% of targets reachable using only linear motion. Several apple picking motions are trialled by Li et al. (2016), with some producing significantly higher rates of fruit damage. Xiong et al. (2020) use push and drag motions to separate obstacles from target strawberries when harvesting. Approach and picking motions may, or may not, translate well from other fruit types to plums. Both the similarity in picking performance between fruit for a given motion, and effective picking motions for plums, remain unanswered questions.

Many types of environmental representation have been used for fruit harvesting, though often these result from other design decisions, rather than being explicitly chosen. Depth information aligned to images is used by Arad et al. (2020) to extract fruit positions as invididual points. They also employ ROS MoveIt! which uses a mesh representation internally. Silwal et al. (2017) likewise combine depth information with imagery, but use a point cloud intermediate representation to encode depth. Circle primitives are applied via a Hough transform to represent fruit in 2D image space.

## 2.4 Detailed Component Studies

Within this thesis a series of component studies are carried out to better explore specific system improvements. These are written as largely self contained sections so

they may be read, or skipped, as desired. In this section literature reviews specific to these studies are presented.

### 2.4.1 Shape Completion for Stochastic Voxel Grids

Incomplete sensor data is a common challenge for robotic grasping, and one which is clearly present for partially obscured trellis obstacles or branches. Object shape completion uses limited geometric knowledge to estimate obscured portions of grasping targets, as in Varley et al. (2017) and Bohg et al. (2011). Many approaches to shape completion exist, such as library matching in Goldfeder et al. (2009); Rennie et al. (2015), object symmetry identification in Bohg et al. (2011); Quispe et al. (2015); Rock et al. (2015), and fitting of parametric primitives like superquadrics as in Vezzani et al. (2017).

These techniques only apply to known or simple object structures. Parametric primitive fitting works well for round fruit, but will fail for more general objects such as plants or farm tools. Deep learning based methods can infer complex shapes and leverage parallel computation for fast results. Work by Wu et al. (2015), applies a deep belief network to simultaneously predict the class, next best view, and completed geometry of an object. Varley et al. (2017) uses a combination of 3D convolutional and two dense reconstruction layers to predict unseen object portions. The input and output of this network are equally sized 3D voxel grids. To perform grasp planning the binarised output grid is converted to a point cloud, upsampled, merged with the input and undergoes a gap removal process followed by marching cubes to transform it into a collision mesh. Choy et al. (2016) take a similar approach but use a long short-term memory (LSTM) layer to fuse multiple viewpoints.

Deep networks can provide accurate statistical representations of their output confidence, traditionally this has been the focus of Bayesian neural networks, as in Mackay (1992). These take the same form as standard neural networks of arbitrary depth but place a distribution, typically Gaussian, over each weight. As shown in Williams (1996), a Bayesian neural network with an infinite number of hidden units is equiva-

lent to a Gaussian process (GP), thus predictive uncertainty can be easily extracted from these networks. One drawback is the additional training complexity required to apply Bayes' rule over the entire weights and biases space, typically computed using Markov chain monte carlo (MCMC) or variational inference, see Andrieu et al. (2003). Recently, Gal and Ghahramani (2016) have shown that model uncertainty can be accurately and efficiently approximated using multiple stochastic forward passes through a standard neural network. The stochastic element is introduced with a stochastic regularisation technique (SRT), typically through dropout layers which are also active at inference time.

Preserving accurate uncertainty estimates for object geometry reconstructions will benefit grasp performance. Many techniques to exploit this uncertainty have been proposed. Dragiev et al. (2013) propose a control law which allows a grasp planner to be biased towards known or unknown regions, leading to an exploration or exploitation style grasp. Another approach is to marginalise the grasp success probability over the posterior distributions of the object representations, grasp success predictors and execution errors, as in Hsiao et al. (2011). They present a fully probabilistic approach which allows the best grasp, given the known uncertainties, to be executed. Although they propose this method for all object representations, the implementation is only explored for database matching reconstruction over a small number of discrete object classes, and for partial point clouds.

This approach of marginalisation over possible object reconstructions is used in the component study of Section 3.2, where each object possibility is sampled from the shape completion network of Varley et al. (2017) using the SRT approach of Gal and Ghahramani (2016). This technique is trained and tested on an existing 3D dataset of objects, but for fruit harvesting, could be applied to predict the stem location when obscured by leaves or branches. Very similar theoretical work to what is presented in Section 3.2 was later simultaneously investigated by Lundell et al. (2019) who reached equivalent conclusions.

### 2.4.2 Monocular Depth Inference in Ambiguous Scenes

An extended literature review for this study can be found in Brown and Sukkarieh (2019). Depth maps are commonly used in robotics for their ability to easily encode and visualise large amounts of geometric information. Inexpensive commercial stereo vision sensors, such as the Intel Realsense and Microsoft Azure Kinect ranges, can be used to generate these, but the texture and lighting assumptions required for these to work effectively are often not met in agricultural environments. For harvesting, the minimum range and physical size of these commercial systems limits their suitability for eye-on-hand sensing. Other methods of estimating 3D geometry, such as time of flight and structure from motion, suffer from similar issues, as reviewed by Blais (2004). Lidar and radar can generate high quality 3D data, but remain too expensive for many agricultural robotics applications.

Direct estimation of a depth map from a single monocular image frame can alleviate these issues, but this is a challenging problem that has only recently seen effective general solutions through the use of deep learning methods, as in Eigen et al. (2014); Laina et al. (2016); Xu et al. (2018); Zhao et al. (2020). If accurate, this technique would allow inexpensive, compact, and robust monocular red-green-blue (RGB) cameras to be used for depth map generation. However, monocular depth estimation is an ill-posed problem when using the pinhole camera model, with multiple physical scenes able to produce identical RGB images, as highlighted in Ladicky et al. (2014).

Two common forms of ambiguity that occur in the pinhole model are scene scale and focal length uncertainty. Suwajanakorn et al. (2015) use multiple frames with a sweep of focal lengths to estimate depth, aperture and true focal length. He et al. (2018) directly feed the focal length to the middle layers of a depth prediction network, with good results on simulated variable focal length images. Work by Ladicky et al. (2014) addresses class bias using a canonical fixed depth plane with dense semantic class pixel segmentation. Assumptions around the object scales for semantically learned classes mean this technique does not address scale ambiguity.

Works such as Ma and Karaman (2018) and Liao et al. (2017) fuse true depth mea-

surements to improve depth map estimation, which also has the effect of eliminating scale uncertainty. Results from the former indicate improved performance over the RGB sensor case only occurs when more than 10 depth samples are used, while the latter requires a lidar sensor for base map generation which is refined by RGB data.

### 2.4.3 Fruit Detection for Harvesting

Fruit detection is a key problem that is common in the literature for goals such as yield estimation, crop health assessment, and harvesting, as used in Bargoti and Underwood (2017a,b); Fernández et al. (2018); Gongal et al. (2015); Koirala et al. (2019); Sa et al. (2017) and Stein et al. (2016). The most basic form consists of placing bounding boxes around each of the fruit in an image. Traditional computer vision methods have been extensively employed, while many recent works make use of deep learning tools. These require large amounts of training data, although this requirement can be relaxed using data augmentation and simulation methods, or transfer learning techniques.

Specular reflections from round fruit, combined with local image gradients, are used in Wang et al. (2013) for object detection under controlled lighting. A support vector machine (SVM) is trained to detect apples using thermal imagery in Feng et al. (2019). The accuracy performance of SVM object detectors has now been surpassed by deep learning approaches, though they remain competitive for classification when using low dimensional hand-engineered image features, as in Kamilaris and Prenafeta-Boldú (2018). Edge features are used with Hough voting and an SVM classifier by Sengupta and Lee (2014) to identify green citrus fruit under varying illumination. Similarly, Maldonado and Barbosa (2016) make use of a bas-relief representation with edges, Hough voting and an SVM to also count green citrus fruit. Nguyen et al. (2016) perform RGB and depth channel thresholding to identify point cloud blobs corresponding to apples. These are separated using a Euclidean distance metric which is tested in the field under semi-controlled lighting conditions.

Harvesting-oriented apple detection data is gathered in Kang and Chen (2020). The

YoloV3, Mask-RCNN and Faster-RCNN architectures are thoroughly tested, along with their own deep learning model, described in Kang and Chen (2019). This implements the idea of focal loss, similar to RetinaNet. Unfortunately the dataset and trained models are not released for comparison and, unlike the study in Section 4.1, the harvesting system does not use an eye-in-hand camera. Gao et al. (2020) train a multi class Faster-RCNN detector to distinguish between different obscuration cases for apples, including those behind branches or wires, so that they can be harvested appropriately.

Gené-Mola et al. (2019) examines apple detection using Faster-RCNN with image, depth, and radiometric data. This multi-modal data was gathered using a robotic platform at a fixed distance from the trellis and results indicated that early depth fusion alone was not effective, but combining image, intensity, and range produced the best detector.

Various sensing modalities for fruit detection, such as hyper-spectral, thermal, and stereo vision are explored in Kapach et al. (2012). Thermal imagery is found to only be useful at certain times of day, while geometry and colour are identified as the strongest features for distinguishing fruit. Similar observations regarding the limitations of thermal imagery use are made in Bulanon et al. (2009), and by Gan et al. (2018) who also present a novel algorithm for fusing thermal and RGB imagery.

Depth data is frequently leveraged for object detection, as in Tu et al. (2020) where a multiscale implementation of Faster-RCNN is improved with the late fusion of red-green-blue-depth (RGBD) imagery. A Microsoft Kinect V2 camera is used, which requires avoiding direct sunlight and is kept a fixed distance from the fruit trellis. This provides detailed and consistent depth data, intended to be used for fruit counting rather than harvesting. Lidar sensors have large depth ranges and very high accuracy, but low resolution. Gené-Mola et al. (2020) leverage the lighting invariance of Lidar to detect fruit in point clouds, captured with and without a commercial air blower being applied to the crop. Combining data both with and without the blower active, led to improved single frame detector accuracy, though it was not beneficial for yield prediction. Late fusion of near infra red and RGB imagery is used to detect a range

of fruit using Faster-RCNN in Sa et al. (2016) and dataset size is found to have a critical impact on detector performance.

A wide range of detectors have been benchmarked on well known computer vision datasets, and many have been individually tested on fruit detection. However, testing and comparison of multiple current generation object detectors on a single fruit detection dataset is lacking. Also unexplored is their performance on eye-in-hand images gathered during the actual harvesting process. The study in Section 4.2 seeks to address these questions.

### 2.4.4 Improving Visual Servoing Using Autocovariance Least Squares

Data filtering methods, such as extended Kalman filtering, are used throughout this thesis and visual servoing is one application of these. To improve filter performance for visual servoing, a study of a noise covariance matrix tuning method is presented in Section 4.4. Improvements presented in this study could allow for more accurate fruit localisation and better motion control during visual servoing. Visual servoing control and autocovariance least squares (ALS) are complex and widely studied topics, for a more in depth literature review of these, the reader is referred to Brown et al. (2020) and Brown et al. (2019).

Visual servoing is comprised of image based visual servoing (IBVS) which occurs in image space, and position based visual servoing (PBVS) in cartesian space, along with combinations of these. Both forms are commonly applied to both robotic manipulation, as reviewed in Corke (1993), and agricultural robotics. For this component study, PBVS is applied to estimate camera pose using a known object position, the converse problem is also common.

Extended Kalman filtering is one iterative estimation technique commonly used for PBVS due to its speed and simplicity. The EKF is optimal for linear systems with Gaussian noise, which is a standard simplifying approximation of camera motion.

However, more complex estimators such as full information estimation (FIE), of which the EKF is a one step case, will perform better for non-linear systems. Rao et al. (2001) explore the constrained version of MHE, a windowed approach to managing the data complexity of FIE. Other recent works by Kong and Sukkarieh (2018a,b); Wallace et al. (2019b) and Wallace et al. (2019a) also address the estimation of model noise parameters and structures for MHE in robotics.

When applying the EKF framework, noise covariance (NC) matrix estimation is a key parameter selection step. Robotics applications often overlook rigorous NC estimation in favour of heuristics or experimentally determined stable values. As presented in Odelson et al. (2006) and further developed in Ge and Kerrigan (2017), the ALS method provides an efficient means of estimating these NC matrices. Unlike the traditional approach of Mehra (1970), ALS provides lower NC estimate variance while guaranteeing unique results and is easily solvable as a convex semidefinite programming optimisation problem. This latter property also allows for constraints in noise structure and positive semidefiniteness to be efficiently imposed, unlike existing PBVS solutions with adaptive noise estimation, such as Janabi-Sharifi and Marey (2010) and Lippiello et al. (2007). Approaches to NC estimation, as reviewed in Duník et al. (2017), also include Bayesian, covariance matching and maximum likelihood, though these are typically less efficient to solve than correlation methods such as ALS.

This study uses a constant velocity linear time invariant (LTI) motion model, as in Wilson et al. (1996) and Assa and Janabi-Sharifi (2015). Such a model can well capture camera motion for small intervals, while being efficient to calculate. Employing the pinhole camera model for the observation function results in a linear time variant (LTV) output matrix and NC matrices are assumed to be constant.

### 2.4.5 Active Perception For Harvesting

Active perception (AP) has been widely explored in robotics, including within agriculture and for harvesting. The general active perception (AP) paradigm consists of planning perception actions to maximise relevant gathered information, using in-

puts from previous perception actions. The active perception component study in Section 4.5 considers the problem of bearings-only fruit localisation, given monocular detections from multiple viewpoints.

Multi-view camera positioning for sweet pepper detection is explored in Hemming et al. (2014); Kurtser and Edan (2018a). The latter reported improved detectability from 50% to 90% for multiple viewpoints, indicating the potential information gains by intelligently selecting viewpoints. A Dirichlet mixture of Gaussian processes is used by Ramon Soria et al. (2018) with Gibbs-Sampling to segment apple clusters. This approach also provides stochastic shape reconstruction for obscured fruit portions.

Harvesting operations using eye-on-hand sensing must consider not just information gain, but also path constraints, such that the gripper finishes at the fruit position ready for picking. Kurtser and Edan (2018b) use an additional economic-cost value function to determine when extra viewpoints are beneficial. None of the above works perform online reactive path adjustment to maximise positional information.

Most relevant to the current study is the work by Mehta et al. (2017) who apply several monocular cameras to a particle filter for 3D fruit localisation. This incorporates a nonlinear motion model and stochastic sensing noise. Cooperative sensing using both a fixed and on-arm camera is used by Mehta and Burks (2014) for citrus harvesting. Controller stability is guaranteed, but experimental accuracy on an artificial tree setup is only sufficient for larger fruit types.

Matching grasp planning tolerances to sensing uncertainty is a simple concept which is often done implicitly in grasp planning systems, but the inverse problem of adapting sensing to suit grasp tolerance is less commonly considered. A gap exists in the literature concerning the application of active perception using a grasp tolerance weighted goal function. Active perception tools allow explicit online optimisation to be applied to this problem. The converse problem of matching gripper tolerance to sensing inaccuracies is explored by Eizicovits et al. (2016), where simulation tools are applied to generate grasp precision maps for two deformable-finger end effectors. Eizicovits and Berman (2014) draw links between grasp affordance density maps and sensing accuracy, but do not consider active sensor control based on these.

# Chapter 3

# Environment Representations & Sensing

Which environmental representation (ER) to use is an important choice when constructing a grasping and manipulation solution. It will inform the sensing type chosen, how grasps are parameterised and planned, plus the forms of useful information available for other tasks like fruit counting. This makes it the first major design choice when constructing the G&M pipeline. To determine an appropriate ER for plum harvesting, several are assessed along with possible sensing modalities.

## 3.1 Environmental Representations

An environmental representation is the mathematical and conceptual model for how sensor information is processed and stored by the system into a useful model of the world. Appropriate ERs are often dictated by the task, and narrow task definitions can benefit from highly specific ERs. A representation for harvesting must capture the fruit locations, obstacles and general trellis position. Considered ERs include Gaussian process implicit surfaces (GPISs), signed distance fields (SDFs) or truncated signed distance fields (TSDFs), meshes, voxel grids or octrees, and geometric primitives.

These have very different strengths and weaknesses. In any representation there is generally a trade-off between compactness and expressiveness. Other considerations include the ease of visualisation, how efficiently sensor data can be processed into an ER update step, and whether a variable or fixed resolution is used.

The representations below were not considered for the reasons listed beside them ...

- Point Cloud. These are often used in systems with lidar input. The variable resolution and sparse nature of point clouds makes them difficult to use with algorithms that expect a fixed input data size, such as many deep learning architectures. Nor are they as memory efficient as more abstract representations.
- Library Indexed Objects. A large library of objects can be constructed offline and a representation built by registering sensor data to library objects and estimating their pose. Agricultural scenes present a near infinite variety of objects with subtle variations, making a library based representation inefficient or impossible.
- Depth Map. These are used as an intermediate step in the ER but are processed into more abstract representations which are easier to work with. Previous work by Morrison et al. (2018) has shown that grasps can be efficiently planned directly on depth maps for top-down picking tasks with high quality depth data.
- Multi-view Maps. Multiple maps of any modality can be combined to provide additional dimensions of representation, such as depth maps from each side of an object. As depth maps are processed into more abstract representations, it is beneficial to fuse multiple views after this step, rather than storing each depth map directly.
- Superquadrics. These are an extension of geometric primitives which allow for modelling of a wider class of shapes. Superquadrics are often combined to build composite object representations made up of multiple primitives. These also possess many of the memory and processing efficiencies of geometric primitives.

Target fruit are easily modelled by regular spheres, and superquadrics are not well suited to modelling branches, leaves or trellis structures, so are not used.

In many cases, the use of rich but difficult to work with representations, such as direct point clouds, is required to employ data-driven methods such as machine learning. This is less beneficial when the perception and planning stages can be separated, as in the presented harvester architecture.

An implicit surface is an object representation constructed using a shape with dimensionality one higher than the object. The object surface is then given by a level set of this surface. GPISs extend this concept by using a Gaussian process (GP) function to learn the higher dimensional shape. This allows Bayesian uncertainty to be properly captured, while elegantly handling unobserved object regions. GPISs are constructed using a continuous kernel function, which limits their ability to capture sharp object features and can induce ringing around edges, as seen near the bottom of the raised section in the Figure 3.1 middle plot.

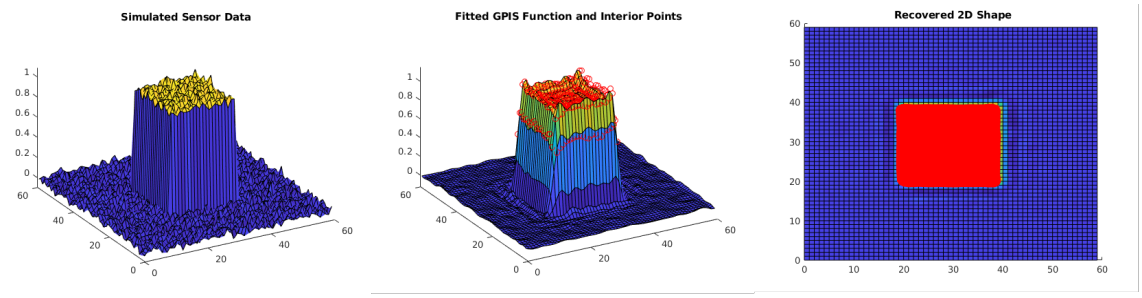


**Figure 3.1** – An example of modelling a 2D square using a 3D GPIS. The left plot shows simulated data with value 1 for points inside a square and 0 for those outside, plus additive noise. The middle plot shows results from a GP fitted to the data and used to predict the z value for each point, points above a threshold of 0.5 are circled red. The right plot shows the same GP function from the top down, regions above 0.5 in height are shaded red.

To generate Figure 3.1 a GP regression function is fit to 3600 data points sampled from $f(x, y)$ over a regular grid from 0 to 60 in each direction. Noise with standard deviation of 0.03 is added. An automatic relevance determination squared exponential kernel function is used with the subset of regressors fitting method. The GP is then

applied to predict $\hat{f}(x, y)$ over the same grid and the 2D shape is recovered by taking a level set at 0.5.

Capturing uncertainty is an attractive feature of GPIS representations and these can be intuitively applied to model round fruit or largely flat trellis walls. However these suffer from inefficient operations during fitting and the additional dimension required for implicit surfaces makes these too computationally demanding to be practical. Training the simple 2D example GPIS for 3600 data points takes 95 seconds.

Signed distance fields are well suited to grasping tasks and are a field given by the distance of that point to the nearest object surface, where points inside an object are negative distances. Constructing these for partially observed objects is difficult, but TSDFs solve this by only considering the signed distance field in a non-truncated region which is close to the zero point of the function, corresponding to an object surface. The projective version of this representation is well suited to projective sensors, such as depth cameras, and can be efficiently stored and compressed. Non-projective TSDFs suffer from very complex updates making the fusion of new sensor data slow. Distinguishing between semantic objects is problematic and typically overlapping TSDFs are required, one per semantic class.

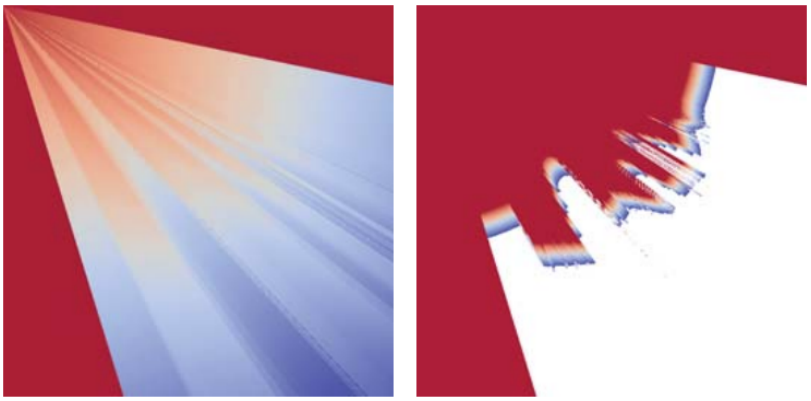

**Figure 3.2** – An example of a projective SDF (left) and a truncated projective SDF (right) from Canelhas (2017). Red values are positive distances from the detected surface, while blue is negative.

Truncated SDFs are considered as an appropriate harvesting choice, and these allow for very simple grasping control laws such as

$$\vec{v} = \kappa \nabla f_{targets} - (1 - \kappa) \nabla f_{obstacles} \tag{3.1}$$

where $\vec{v}$ is the end effector velocity, $\kappa$ is an obstacle avoidance tuning parameter, $f_{obstacles}$ and $f_{targets}$ are the obstacle and grasp target SDFs respectively. This attempts to place the gripper on the closest target object surface while avoiding nearby obstacles, but will fail with local minima and is entirely myopic.

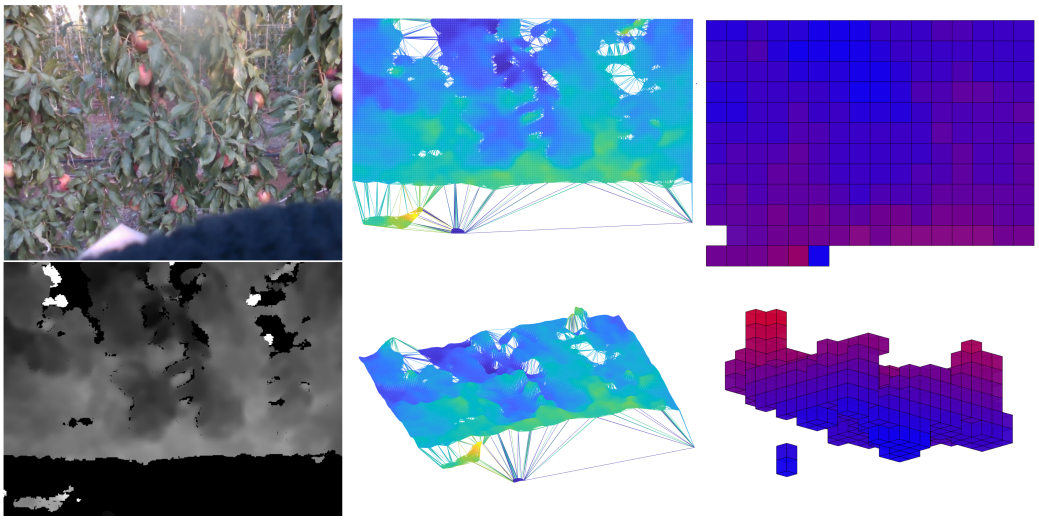

**Figure 3.3** – An example of raw sensor data in RGB and Depth (left), a mesh representation of this (middle) and a voxel grid representation (right). Artefacts such as missing depth values lead to poor mesh segments which require post-processing. The voxel grid is much more compute intensive when performing visualisation and manipulation operations but provides a better volumetric representation of the data.

Meshes are the dominant representation in computer graphics and are frequently used in robotic scene representation as well. Typically, meshes are encoded as a set of vertices, plus facets which contain these. This allows fast access to object properties such as surface normal, while powerful and efficient tools developed for computer graphics can be used to interact with object meshes. Capturing geometric uncertainty is difficult to do using meshes, making them unsuitable for modelling the hard and soft obstacles present during harvesting. However, for rigid bodies such as the robot and trailer platform, these are a compact and efficient choice. Figure 3.3 shows raw

sensor data of a trellis encoded as both a mesh and voxel grid. The connected nature of the mesh leads to poor results where there are discontinuous jumps in the depth map. Creating a closed, or 'watertight', mesh is required to properly represent 3D objects. In this case, the trellis is well captured as a non-closed surface mesh.

Voxel grids divide the environment into 3D cubes known as voxels. Each of these can have arbitrary associated properties, such as the presence of an object or visibility of that region to a sensor. Voxel grids are easy to visualise and can be readily transformed into meshes or TSDFs for grasp planning. Being a dense representation, voxels are very inefficient. Octomaps (octal maps) try to address this using a variable resolution octal tree data structure to model large unoccupied regions with large cubes and areas of high complexity using many smaller cubes.

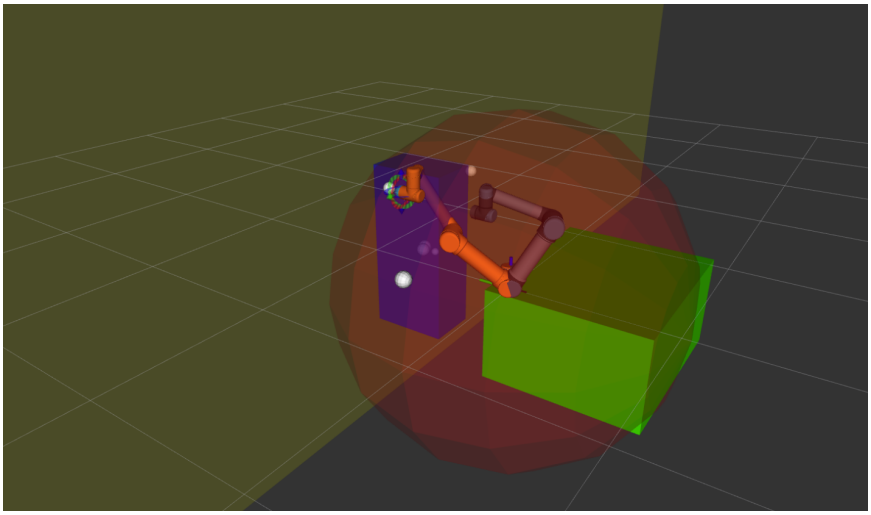

**Figure 3.4** – A frame from the picking experiments showing the actual environmental representation used. The arm, planned pose and trailer are stored as meshes for motion planning (textured, orange and green shapes). Target fruit are modelled as sphere primitives and visualised as meshes (white). Rather than constructing obstacle regions from sensor data, a simplified representation defined by an obstacle plane primitive is used (yellow-green). The current picking area is also shown for convenience (blue) along with the arm workspace (red).

Cube shaped voxels are not good for modelling round fruit, or the fine detail of branches and leaves. Figure 3.3 shows a voxel encoding of a trellis scene with very little detail being obvious at that resolution level. Approximately 78% of the voxels in that

scene go towards modelling free space, showing the importance of using octomaps.

The final system design uses a composite of sphere primitive, mesh, and planar representations for the various tasks required, shown in Figure 3.4. Spherical plum fruit are modelled as geometric sphere primitives which are compact and accurate. Rigid obstacles, including the robot, trailer and gripper are modelled as meshes. Specific assumptions are made when modelling the hard and soft trellis obstacles, these are further described in Section 5.2. Under these assumptions the trellis obstacles can be represented with 2D plane primitives for planning. All objects are converted to meshes for motion planning and visualisation.

## 3.2 Study: Shape Completion for Stochastic Voxel Grids

Shape completion is required when only part of an object is observed and the obscured geometry must be inferred. This is a common problem in robotics, where projective sensors are used and actions should be performed without exhaustively scanning the scene first. For harvesting, this could be applied to predict the extent of partially obscured trellis obstacles or branches. Deep learning methods are effective at predicting shape completions given partial data, but are difficult to use with variable resolution ERs. Voxel grids do not have this issue and are a natural fit for deep neural network (DNN) shape completion methods. However, in most voxel encodings, as in Figure 3.3, the occupancy of a region takes on a binary value without properly capturing sensing and modelling uncertainty. This uncertainty in object pose or geometry is important for generating good grasps or planning for obstacle avoidance. To address the issue of capturing uncertainty when performing shape completion in voxel grids, a stochastic voxel grid representation is investigated and a method for marginalising predicted grasp performance over this representation is presented.

In this study, a depth camera is applied to a simulated indoor scene to generate input binary voxel maps with incomplete geometry. The neural network of Varley

et al. (2017) is extended with runtime dropout layers, allowing for multiple object reconstructions to be sampled. Grasp planning is performed using these multiple samples, allowing grasps to be marginalised over probable object shapes. An indoor setting with common objects is chosen because standard datasets already exist for performance comparison in this setting and gathering full object geometry ground truth for trellis environments is very challenging.

### 3.2.1 Method

The input for shape completion is chosen to be a $40 \times 40 \times 40$ binary voxel grid for one view of an isolated object, resulting in a partial 3D model. This simulates a 3D sensor observing an unknown object and representing the output as a voxel grid. The voxel grid resolution is determined by GPU memory limitations. A deep convolutional neural network (CNN) is applied to perform shape completion by predicting the occupancy of obscured voxels. Typically, uncertainty is not well represented by CNNs however, as shown by Gal and Ghahramani (2016), deep network model uncertainty can be approximated using multiple stochastic forward passes through that network. The stochastic element is introduced by stochastic regularisation techniques (SRTs), such as through standard dropout layers which are also active at inference time. By applying this technique to the shape completion CNN from Varley et al. (2017), voxel wise uncertainty is extracted for the inferred occupancy of obscured voxels. Gathering ground truth data for harvesting relevant scenes is challenging, so an existing dataset of isolated objects is used instead.

Each forward pass of the network with dropout enabled results in one stochastic object reconstruction (SOR). This is a sample from the probability distribution of possible object reconstructions learned by the CNN. These independent SORs can be used with standard grasp planners such as the OpenRave grasping module by Diankov and Kuffner (2008), which samples grasp approach directions uniformly over a box around the SOR. The OpenRave simulator then performs a grasp by moving the hand towards the SOR centre until contact and closing the fingers. Force closure is

checked and a grasp quality metric (GQM) corresponding to the minimum contact distance from the object centre is computed for each grasp. This allows grasps to be planned and ranked for each SOR, the known approach directions means the same grasp can be applied to multiple SORs. The grasp with either the highest mean or minimum quality over all SORs is selected, effectively marginalising over the expected object geometry.

### Stochastic Voxel Model

The stochastic voxel grid is a representation of object geometry using a $40 \times 40 \times 40$ grid of cells. Observed voxels take a value of 1 if any part of an object intersects that voxel, and 0 otherwise. The occupancy of each unobserved voxel is modelled as a Bernoulli random variable

$$\begin{aligned} p_o &= p_{occupancy}(V_{x,y,z}) \\ p_o &\sim \text{Bernoulli}(\hat{p}) \end{aligned} \tag{3.2}$$

where $V_{\text{x,y,z}}$ denotes a voxel at x, y, z, the expected value and variance of the occupancy estimate are given by $\mathbb{E}(p_o) = \hat{p}$ and $\text{Var}(p_o) = \hat{p}(1 - \hat{p})$. For highly certain voxels $\hat{p}$ takes a value close to zero or one, highly uncertain voxels will have $\hat{p} \approx 0.5$.

### Stochastic Regularisation as Bayesian Approximation

Previously, Gal and Ghahramani (2016) have shown that a standard neural network (NN) with dropout applied before all of the weights layers approximates a deep GP. In practice, the predictive mean and variance of a given model can be approximated directly using the results of multiple stochastic forward passes, added to the inverse model precision.

$$\text{Var}[p_o] \approx \text{Var}[\hat{\boldsymbol{p}}_{\boldsymbol{o}}] + \varphi^{-1} \tag{3.3}$$

where $\hat{\boldsymbol{p}}$ is a random vector of the stochastic network output for $M$ forward passes and the inverse model precision is given by

$$\varphi^{-1} = \frac{2N_d\lambda}{p_{keep}\, l^2} \tag{3.4}$$

where $N_d$ is the number of training datapoint pairs, $\lambda$ is the weight decay used in $L_2$ regularisation, $p_{keep}$ is the probability that a given connection will not be dropped by the dropout layer and $l$ is the prior length scale which is a data based hyperparameter that comes from the GP interpretation of dropout. All results presented in Section 3.2.2 use an inverse model precision of 0.145 which was empirically tuned using the first 500 examples from the testing dataset.

### Polynomial Loss Function

The standard cross entropy loss function for binary classification, equivalent to the Bernoulli cross entropy with the known labels, is given by

$$\text{Loss} = -\tau \log(\hat{\tau}) - (1-\tau)\log(1-\hat{\tau}) \tag{3.5}$$

where $\tau$ is the true class label and $\hat{\tau}$ is the predicted label. In place of this, a simple polynomial loss function is proposed for testing, which is motivated by the need to calculate a loss for both regression and classification. Because the latter of these is measured using binarised outputs according to

$$\text{Voxel Class} \begin{cases} \text{Class} = \text{Occupied}, \tau = 1 \text{ if} & \hat{p} > 0.5 \\ \text{Class} = \text{Unoccupied}, \tau = 0 \text{ if} & \hat{p} \leq 0.5 \end{cases} \tag{3.6}$$

The log function cannot be used with this. Instead, a simple polynomial loss is used, given by

$$\text{Polynomial Loss} = \frac{1000}{N_V} \sum_{V_{x,y,z}}^{V_{X,Y,Z}} (\hat{p} - p)^{\varrho} \tag{3.7}$$

Which is the mean of the difference between the voxel wise predictions $\hat{p}$ and labels $p$, raised to the power of $\varrho$. $N_V$ is the number of voxels, with a numerator of 1000 used for notational convenience. The value of $\varrho$ is a hyperparameter which sets the aggressiveness of the network in assigning binary class values close to 0 or 1, rather than uncertain regression values. This must be even for a symmetrical loss function and $\varrho = 4$ is used.

### Model Architecture & Dataset

The model architecture is modified from Varley et al. (2017) by adding a dropout layer with keep probability 0.5, to the final densely connected layer, as shown in Figure 3.5. All layers prior to the final dense layer have their pretrained weights frozen and the final reconstruction layer is retrained with dropout in place using Keras and a Tensorflow backend. This preserves the learned features from the 3D convolutions. Unlike the original network, the resulting output is not thresholded and remains as a float value between 0 and 1.

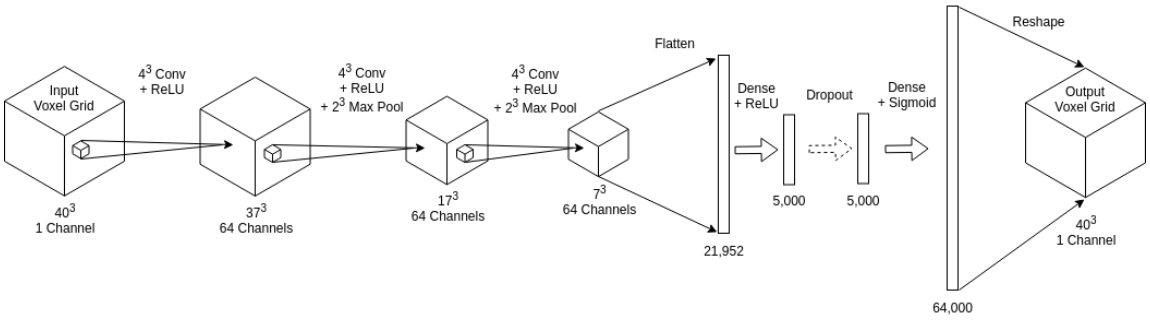


**Figure 3.5** – The shape completion network with an additional stochastic dropout layer which is active at inference time.

The complete training and testing dataset is that used in Varley et al. (2017) which is generated using the YCB data from Calli et al. (2015), and Grasp Database from Kappler et al. (2015). It consists of binary voxel grids of 726 uniformly sampled views,

for each of 608 objects taken from the YCB and Grasp Database datasets. Following their methodology, this is split into a training dataset of 486 objects and a test set of 122 objects. See Figure 3.8 for example inputs. With fine tuning from pretrained weights, the dropout enabled model loss stabilises after approximately 3.6 hours of training on an NVIDIA K80 GPU.

### Model Validation

The stochastic reconstruction network is validated against the state of the art network in Varley et al. (2017). Because a low numerical loss provides no guarantee that each SOR will resemble a possible object or be contiguous, 50 stochastic reconstructions each, of 4 representative objects are also checked visually. For the test set of 122 test objects, 88,572 distinct binary voxel grids are generated, one for each view. A randomly selected set of 500 are used to determine the inverse model precision and the remainder are used to calculate the non-stochastic and stochastic losses.

A sweep over the number of stochastic object reconstructions to generate, was conducted by stepping from 10 to 200 network passes in increments of 10. Mean loss was unaffected by the number of passes, while the variance of this loss continues to decline with increasing passes. Therefore, 50 passes was chosen as a trade-off between performance and reduced variability. A set of 4 challenging objects were selected to test the grasp performance of this method and to illustrate the reconstruction results. These are a detergent bottle, stapler, spraybottle, and lightbulb, as shown in Figure 3.6.

Two forms of stochastic loss are reported. The *mean stochastic* loss empirically calculates the $\hat{p}$ of each voxel as the mean occupancy value over the 50 reconstructions. While *variance stochastic* applies Equations 3.3 and 3.2 with the empirically calculated sample variance and model precision to recover $\hat{p}$.

### Grasp Performance Validation

Following confirmation that the dropout network produces reasonable SORs, these are used in a grasping simulation to assess whether planning over multiple possible

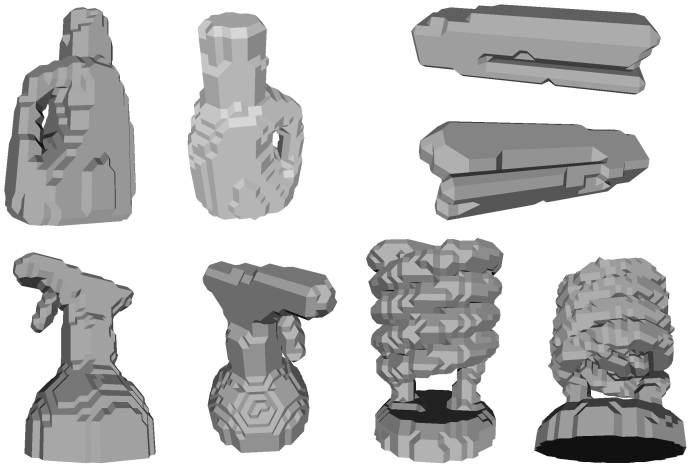

**Figure 3.6** – The four illustrative test object ground truth completions. Clockwise from top left; detergent bottle, stapler, lightbulb and spraybottle.

reconstructions leads to a more robust grasp than a single reconstruction. By generating multiple SORs it is possible to sample from the object geometry distribution learned by the CNN, which accounts for the correlation between voxels, unlike directly sampling from the Bernoulli distribution of each individual voxel.

A model of a three-fingered reconfigurable gripper, further described in Section 5, is utilised in OpenRave to plan grasps which are assessed using the inbuilt grasp performance metric of Diankov (2010). This is calculated as the sum of the squared Euclidean distances between each contact and the bounding box centre

$$\text{Performance} = -\left[\sum_{C_{x,y,z}} (C_{x,y,z} - T_{x,y,z})^2\right] \tag{3.8}$$

where $C_{x,y,z}$ are the contact x, y, z coordinates, $T_{x,y,z}$ are the target object axis aligned bounding box centre coordinates. Higher values indicate better grasps. The grasping scenario is shown in Figure 3.7, with the first detergent bottle SOR on a table in front of the hand, which is mounted to a standard Barrett WAM-segway platform.

Algorithm 3.1 describes how planned grasps are marginalised over multiple SORs for a single object. First a set of grasps is sampled on a sphere around the object location

**Algorithm 3.1:** The Uncertainty Aware Shape Completion Grasp Planning Algorithm

**Input:** partial_voxel_grid, object_pose, hand_model
**for** 50 passes **do**
  SORs *at* pass ← run SRT CNN
**end for**
mean_SOR ← voxelwiseMean(SORs)
*load* mean_SOR,hand_model *to* OpenRave *at* object_pose
grasps ← generateGrasps(mean_SOR, hand_model)
valid_grasps ← validateGraspsForIK(grasps)
**for** *each* SOR *in* SORs **do**
  *load* SOR *to* OpenRave *at* object_pose
  scores ← getGraspPerformance(valid_grasps, SOR)
**end for**
top_mean_grasp ← valid_grasp *at* max(mean(scores))
top_min_grasp ← valid_grasp *at* max(min(scores))
**Output:** top_mean_grasp, top_min_grasp

and checked for reachability. Each inverse kinematics (IK) valid grasp is then tested on the 50 reconstructions, where both the minimum and mean performance for each grasp, over all SORs, is considered. Grasp quality on the true object geometry is also calculated using the ground truth voxel grid.

### 3.2.2 Simulation Results

#### Model Validation Results

Sample input data and the corresponding mean reconstruction are shown in Figure 3.8. To voxelise the output, the mean of 50 stochastic passes is calculated for each voxel and thresholded at $\hat{p} > 0.5$. Table 3.1 reports the three forms of polynomial loss. The non-stochastic network type has previously been shown in Varley et al. (2017) to produce good reconstructions, so forms a baseline for comparison.

The mean stochastic approach performed best under this loss function, while variance stochastic unexpectedly did worse than the baseline. This may indicate that the

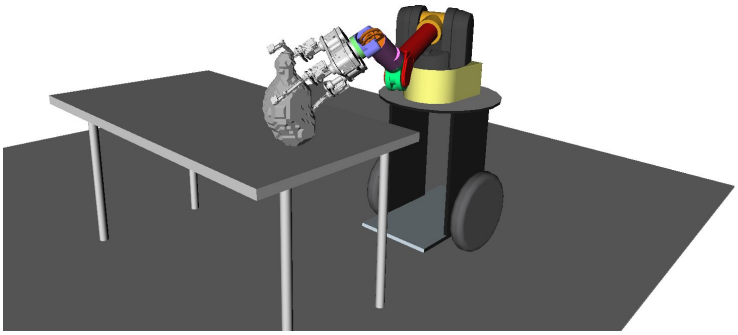

**Figure 3.7** – The OpenRave grasping scenario showing a detergent bottle reconstruction.

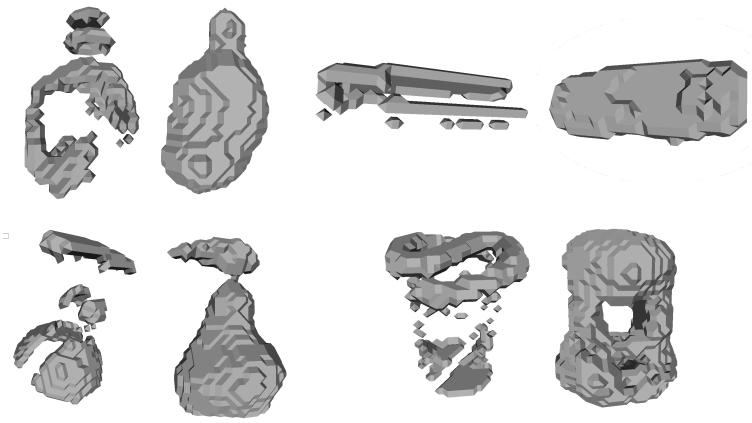

**Figure 3.8** – Four representative voxel grid inputs (left of pair) and their corresponding mean stochastic reconstructions (right of pair), each of these is the voxelised mean of 50 SORs.

model precision is not well estimated, or the Bernoulli probability relationship in 3.2 is not a good choice of voxel model. Inter-object variance is quite large with certain complex geometries resulting in poor reconstructions with large holes and occupied voxels outside the true object extents, 95% of objects have a mean stochastic loss value less than 20.58 while the maximum is 195. The worst performing reconstruction was a bowl with the concave half facing directly away from the viewpoint, which was incorrectly reconstructed as a sphere. A lack of non-spherical objects with this partial appearance in the training set, led to a confident but incorrect prediction, highlighting

the importance of a diverse training set.

| | **Non-Stochastic** | **Mean Stochastic** | **Variance Stochastic** |
|---|---|---|---|
| **Mean** | 7.964 | 7.352 | 9.772 |
| **Std. Dev.** | 13.475 | 12.534 | 9.724 |

**Table 3.1** – Polynomial loss results for each reconstruction method on test set.

The network inference runtime is an average of 77.4ms per pass, with 50 passes per object, on an Nvidia GTX 860M requiring 4GB of memory. SOR passes can be executed in parallel to reduce time requirements. Inference at around 13Hz should be possible during harvesting, with additional GPU hardware.

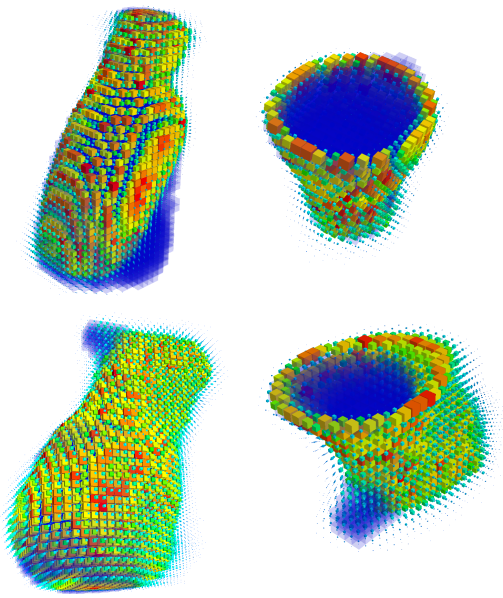

**Figure 3.9** – Voxel uncertainty for reconstructions of the detergent bottle (top left) and an inverted vertical slice of this (top right). Also for the spraybottle (bottom left) and an inverted slice of this (bottom right), the blue protrusion is the rear of the spray head.

For two of the illustrative objects, the voxel wise uncertainty is visualised in Figure 3.9, for the entire object and a slice through the mid-plane. The size and colour of each voxel is scaled by $1 - |1 - 2\hat{p}|$ so larger voxels are more uncertain. Ground truth reconstructions are in blue. The uncertain voxels are correctly clustered around the

surface of the object, and are more prominent in complex unobserved regions, such as the rear of the spraybottle head. The interiors of both objects are high confidence regions.

Combined with visual inspection of the 200 SORs produced for the objects in Figure 3.6 it is clear that the CNN provides coherent and reasonable object reconstructions. Performing stochastic passes with this CNN is an effective means of sampling from the distribution of geometries learned by the network and conditioned on the voxel grid input.

### Grasp Performance Results

Performance of the marginalised grasps by the criteria of mean and minimum score is presented in Table 3.2. Mean grasp performance is calculated by finding the best grasp for each of the 50 SORs, applying these to the true object and averaging the resulting scores. True performance of the best mean, and min grasps refers to the *top_mean_grasp* and *top_min_grasp* quality respectively, found using Algorithm 3.1, and tested on the true object shape.

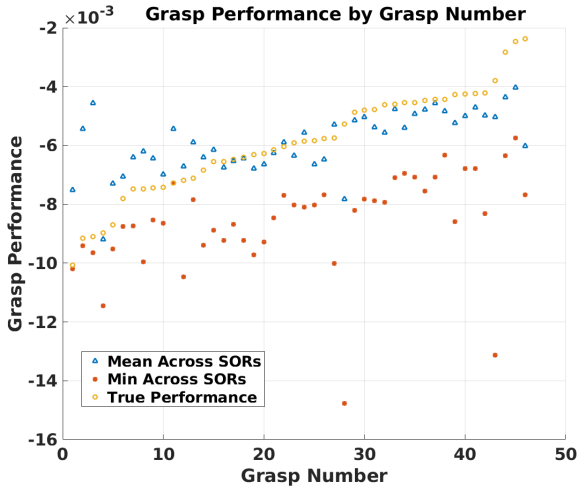


**Figure 3.10** – The performance of each valid grasp generated for the detergent bottle, with mean and min calculated across all reconstructions, ranked by the true grasp performance metric on the ground truth object shape.

| | **Detergent** | **Stapler** | **Spray-bottle** | **Light-bulb** |
|---|---|---|---|---|
| No. of Generated Grasps | 864 | 2320 | 960 | 1704 |
| No. of IK Valid Grasps | 46 | 7 | 33 | 73 |
| Mean Grasp Performance | -0.0059 | -0.0042 | -0.0130 | -0.0186 |
| True Performance of Best Predicted Mean Grasp | -0.0025 | -0.0024 | -0.0031 | -0.0163 |
| True Performance of Best Predicted Min Grasp | -0.0025 | -0.0024 | -0.0037 | -0.0171 |
| Average Percentage Improvement Of Best Mean Grasp Over Best Individual SOR Grasps | 85.6% | -6.7% | 68.4% | 4.5% |

**Table 3.2** – Stochastic reconstruction marginalised planning samples and grasping performance results (higher performance is better).

Selecting the best grasp for a single SOR would, on average, perform 38% worse than considering grasps marginalised over multiple SORs, although the opposite is true for the stapler. Note that only a small number of valid grasps were generated for this object despite doubling the sampling density, this is due to its small size making grasp planning challenging. The average testing time per grasp is 290ms, though the process could be fully parallelised.

Figure 3.10 shows the 46 valid grasps for the detergent bottle, ranked by true performance. With 50 SOR samples, the minimum predicted quality of a given grasp over all reconstructions, did empirically lower bound the true performance of that grasp. The mean grasp quality correlates closely with true grasp performance, indicating that the true object reconstruction is accurately captured by the aggregated SORs.

Figure 3.11 shows the *top_mean_grasp* performance as a red line, along with the best grasp performance per individual SOR. In this case, all SOR grasps were worse than or equal to the marginalised grasp, although this was not true for other example objects. This indicates that the best grasp marginalised over all SORs is typically

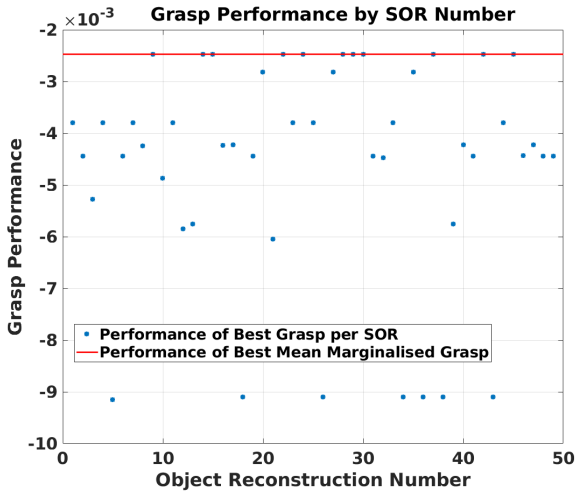


**Figure 3.11** – The true performance of the best grasp generated for each detergent bottle SOR, and the true marginalised grasp performance (higher is better).

also the best grasp for each individual SOR, though not necessarily vice-versa.

One limitation of this approach is that the probability of a given reconstruction being correct is not explicitly represented, instead it is built up from the appearance frequency of SORs similar to that geometry in the object reconstruction set. This requires a large number of SORs to properly approximate the underlying distribution of reconstructions produced by the CNN. However, many methods exist for improving sampling efficiency, such as importance sampling.

### 3.2.3 Stochastic Shape Completion Study Conclusion

Applying stochastic regularisation techniques to the reconstruction network does capture voxelwise uncertainty. Sampling from this network results in sensible object reconstruction possibilities, which preserve the correlations between voxels. This allows multiple possible object reconstructions to be tested during grasp planning, resulting in better performing grasps which are marginalised over reconstruction uncertainty. While there are well known limitations to grasping simulation and the accuracy of grasp quality metrics, marginalisation over 50 SORs improved simulated performance by an average of 38% for the 4 tested objects.

While the applied method builds upon cutting edge shape completion work, the result of this when reconstructing isolated and perfectly sensed objects with common and easy to model geometries is still not highly accurate. If applied to the much harder problem of reconstructing multiple fruit, plant and trellis objects in a scene with complex organic shapes, this method is unlikely to yield useful results for the G&M task. For this reason, and based on the inefficiency and poor suitability of using a voxel grid ER, stochastic voxel grid shape completion was not applied to the final harvesting system. With additional development and training datasets specific to tree crop tasks, this technique may become viable for use in harvesting. Predicting the geometry of trellis and tree branch obstacles from partial views remains a useful functionality in harvesting and improved shape reconstruction is one possible pathway to this.

## 3.3 Sensor Selection

A combination of meshes for obstacle motion planning and geometric primitives for target tracking is selected as the environmental representation. The appropriate platform sensors can then be chosen. Criteria when selecting these include compatibility with the ER, range, resolution, accuracy, mass, power requirements, size, and robustness to environmental conditions. Several forms of sensing were considered, as listed below. The term 'embedded' here refers to those integrated into the robotic end effector for harvesting.

- 2D RGB cameras. Including embedded micro-cameras, high resolution cameras and event cameras.
- 3D cameras. Including stereo, time of flight and structured light cameras.
- Lidar. Including mechanically scanned and solid state.
- Tactile and embedded. Including fingertip deformation sensing, surface texture sensing, and embedded force-torque sensors.

- Pre-touch. Including sea shell effect, embedded time of flight and capacitive.
- Multi-spectral and thermal imaging
- Soft robotics. Including pressure, deformation and vibration sensors.

Tactile, pre-touch and soft robotic sensors were ruled out based on a complexity to benefit trade-off. These may be used to address specific problems identified in the harvesting procedure, but are complex and difficult to integrate so were not included in this prototype. 3D cameras combining stereo vision and structured light were chosen in place of lidar, these are much more compact and lower cost. The disadvantage is shorter range and lower accuracy, which is partially corrected for using the filtering described in Chapter 4. Susceptibility to outdoor lighting conditions is the primary drawback of cameras, as seen in Section 4.1. Modern DNN object detection algorithms were expected to perform well enough that using multi-spectral cameras for fruit detection would not be sufficiently beneficial to justify their additional cost.

Three key functionalities were identified for the sensor selection; a 3D camera to localise fruit from a distance, a wide angle embedded 2D camera for final approach control which avoids obscurations, and a platform tracking camera to provide the trailer pose in a world frame. Specifications for these are summarised in Table 3.3.

| **Name** | Realsense D435i | Wide Angle Camera | Realsense T265 |
|---|---|---|---|
| **Type** | RGBD Camera | RGB Camera | SLAM Camera |
| **Purpose** | Fruit 3D localisation | Final approach IBVS control | Platform tracking in world frame |
| **Mass (g)** | 72 | 12 | 60 |
| **Size (mm)** | 90 x 25 x 25 | 25 x 24 x 18 | 108 x 25 x13 |
| **Resolution (px)** | 1280 x 720 | 1080 x 1920 | 848 x 800 |
| **Field of View (Diagonal °)** | 77 | 160 | 163 |
| **Range** | 10m | N/A | Acceleration: ±4g ±2000 Deg/s |
| **Accuracy (Relative Error)** | <2% | N/A | <1% |
| **Framerate (Hz)** | 30 | 30 | 200 |

**Table 3.3** – Selected imaging sensor specifications.

RGBD cameras provide semi-dense depth maps which integrate well with the spherical fruit ER, most lidar models lack the sampling density to easily reconstruct spheres.

Simultaneously, the additional structure imposed by the geometric primitive ER allows a lower accuracy sensor to still provide effective information. A Realsense D435i was chosen as the RGBD camera, this uses a near infrared imaging pair with a speckle pattern projector to increase the texture on homogeneous surfaces. It also integrates an RGB sensor to provide an aligned RGBD image, the onboard inertial measurement unit (IMU) is not used. The D435i is too large to fit within the soft gripper, so is mounted on top, as shown in Figure 3.12. Another key drawback is the narrow field of view (FoV) which means it cannot be used for controlling final fruit approach and grasp execution. To overcome this, a separate wide angle RGB camera is situated within the soft gripper cup. All fingertips are visible in this camera, right up until the point of fruit contact. Being situated coincident with the z axis of the gripper also reduces the probability of target obscuration by leaves.

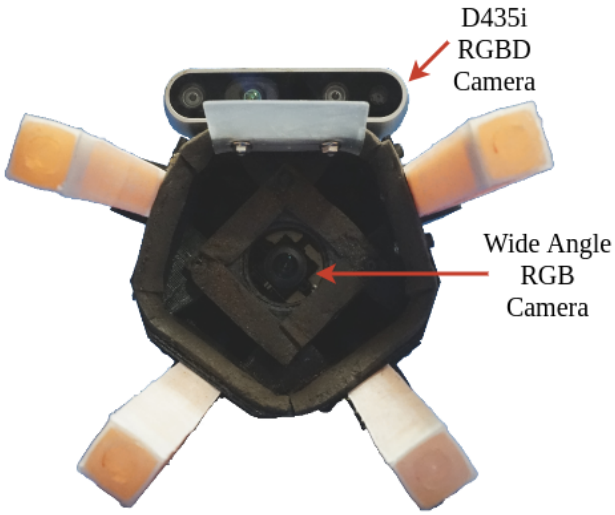


**Figure 3.12** – The two primary sensors used for the system, consisting of a D435i RGBD camera and wide angle RGB camera. Shown in their final mounting locations on the soft gripper, as described further in Section 5.1.2

Many existing harvesting designs use a fixed camera operating in a global frame for perception. Being a first effort for plum crops, an eye-on-hand design was chosen. Increasing image information as the gripper approaches the target, which can be used for visual servo control, is one benefit of this. But the primary reason was

camera positioning flexibility during initial research, as a mobile camera can effectively simulate a fixed camera within the arm workspace, if that is determined to be more accurate

While harvesting can all be planned in a local frame, tracking fruit in a world frame allows for yield mapping, growth tracking and fruit counting. All of these are valuable tools for growers, which are built upon the data required for autonomous harvesting. To track the trailer relative to a world frame, the Realsense T265 simultaneous localisation and mapping (SLAM) camera is chosen. This visual SLAM solution includes a wide angle stereo imaging pair, IMU and onboard processing to allow for SLAM loop closure when estimating the trailer pose within the orchard.

Ideally the wide angle camera used for the final approach controller would also provide 3D data. Embedding a second camera of that size, to form a stereo pair, is not possible. Custom designed optics could achieve the required compactness, but with a significantly increased development timeline and additional platform cost. Instead, two algorithmic approaches are investigated for generating 3D information from this camera. The first applies a deep network to directly predict a depth map from a single RGB image, this relies on learning the typical relationships between object classes, features, optical phenomena, and geometry. The second uses an active perception framework to estimate fruit positions using multiple views and then optimise the camera trajectory to reduce position uncertainty, and is presented in Section 4.5.

## 3.4 Study: Monocular Depth Inference

With depth maps being a valuable scene representation and monocular cameras the most flexible sensing modality, the combination of these is a natural topic for exploration. While generating 3D data using multiple camera sensors or poses is a well studied topic, the generation of depth maps from a single monocular camera frame is a much harder problem which has only recently been effectively approached using deep learning techniques. Monocular depth inference could be used with the wide angle RGB camera to provide depth maps within the canopy during harvesting. Structure

from video is an alternative, but performs poorly when frames are not continuous, such as when passing through leaves. Stereo vision can be used, as in the D435i camera, but requires a larger and more expensive camera.

Therefore, predicting accurate depth maps using images from the wide angle camera would allow for better grasping accuracy and expand the system sensing capabilities. It should be noted that some applications use relative depth maps, where true scale is ignored, but absolute depth maps are required for robotic manipulation. This study section briefly summarises key methods and results from Brown and Sukkarieh (2019) which are specific to harvesting. Further details and original figure sources are available in that paper.

Monocular depth inference is a challenging and ill-posed problem due to scene scale and focal length ambiguity. It requires learning correspondences between object features, geometry, and projected images. Within agriculture, object classes are typically well known, but geometry is continuously variable as plants grow and change. This causes learned object geometries to be ambiguous with respect to the true object scale. An analogy is viewing a tree from a distance or a sapling from close proximity, with both producing similar images.

Existing work has shown monocular depth inference to be a tractable problem when using large training sets from consistent scenes, such as indoor environments or autonomous vehicle data. But issues of scale ambiguity are not well addressed. Experiment results indicate that addition of a single true range measurement is sufficient to resolve simulated scale ambiguity in two existing datasets. It was also determined that the projection method used when sampling range points from a depth map is important to inference performance. Limitations in the range sensor chosen were observed when applying this to outdoor data collection. Fine tuning for agricultural scenes, on models trained using an indoor dataset, was not found to work, so large RGBD agricultural datasets would be required to apply this technique in practice.

### 3.4.1 Background

#### Pinhole Camera Model

The pinhole camera projection model is used to define correspondences between the depth map, colour image and 3D scene points. The distortion model from Heikkila and Silven (1997) is assumed to apply. Two axes systems and a point are defined in Figure 3.13. $(X^C, Y^C, Z^C)$ is the camera frame, where the pinhole aperture is located. $(u^C, v^C)$ are the image plane axes, separated from the camera frame by the focal length. $P^C$ is a scene point in the camera frame and $\boldsymbol{P^C}$ is a point in the $u, v$ image plane.

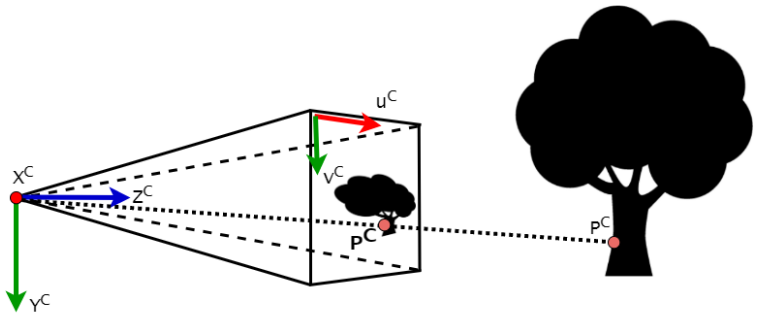


**Figure 3.13** – The pinhole camera model showing the camera focal point and image plane. Also, the projection of scene point $P^C$ into the image plane $(u^C, v^C)$ at $\boldsymbol{P^C}$.

Homogeneous coordinates are used to allow linear projection operations given by

$$\begin{bmatrix} \boldsymbol{P^C}_u \\ \boldsymbol{P^C}_v \\ 1 \end{bmatrix} = \mathbf{C} \begin{bmatrix} P_X^C \\ P_Y^C \\ P_Z^C \\ 1 \end{bmatrix}, \tag{3.9}$$

where, $\mathbf{C}$ is the intrinsic camera matrix which is defined as

$$\mathbf{C} = \begin{bmatrix} f_x & c & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.10}$$

$u_0, v_0$, known as the principle point or optical centre, is where the projection of a point on the $Z^C$ axis falls in the $u^C, v^C$ frame. The image plane axis skewness is denoted $c$, and is assumed to be zero, while $f_x, f_y$ are the focal lengths in $X^C$ and $Y^C$.

Figure 3.14 illustrates a range sensor mounted parallel to, and below, the camera. $(X^S, Y^S, Z^S)$ is the location of the range sensor which returns a reading of the $Z_S$ distance. Each scene depth map is defined within the frustum formed by the image plane projection and maximum depth map value. The depth map value for point $P$ is the $Z^C$ distance to this point, which is equal to $Z^S$ as these are aligned. Note that the depth map is defined by the perpendicular, rather than radial camera distance to the closest object point falling in that depth pixel.

To generate simulated range measurements using ground truth depth maps, it is required to find the depth map pixel with the smallest distance $\varrho$ which satisfies

$$P^C = \begin{bmatrix} P_X^S \\ P_Y^S \\ \varrho \end{bmatrix} \tag{3.11}$$

$$\varrho \geq \text{Depth Map}\left(\boldsymbol{p^C}_u, \boldsymbol{p^C}_v\right)$$

$$\boldsymbol{p^C}_u = \frac{f_x P_X^S}{\varrho} + u_0$$

$$\boldsymbol{p^C}_v = \frac{f_y P_Y^S}{\varrho} + v_0$$

where $P_X^S$ and $P_Y^S$ are the range sensor location in the camera axes. This process corresponds to projecting each of the possible range measurements along ray $Z^S$ and finding the first distance that falls behind the depth map value at the depth pixel intersected by $Z^S$.

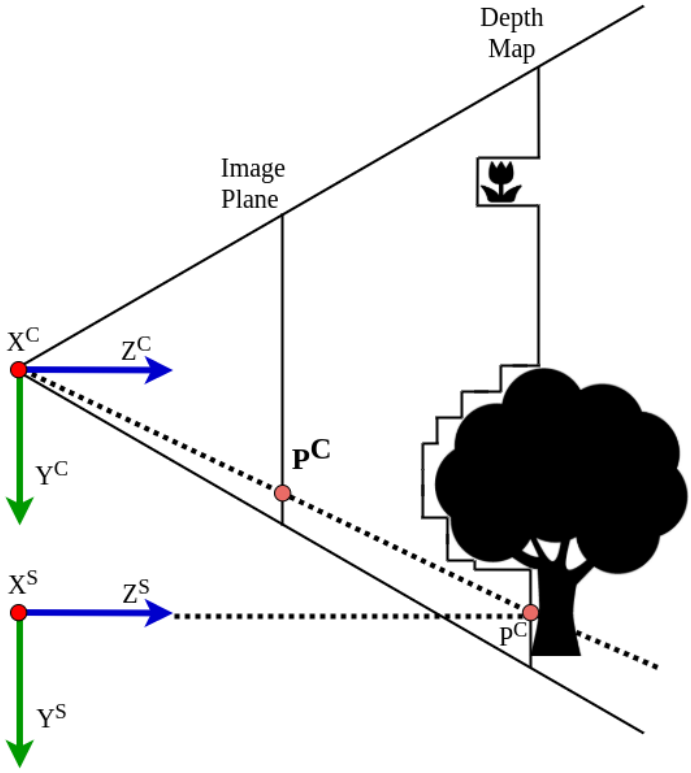


**Figure 3.14** – A 2D illustration of the axes locations for the camera and range sensor. A single point on the depth map and its image plane projection is also shown. Note the depth map is defined by the $Z^C$ distance.

### Scene Depth Ambiguity

Projecting a point from the image plane into a scene using Equation 3.9 results in a ray. Because 3.9 provides 2 equations constraining 3 variables, additional information must be provided to recover true scale projections. If the point is part of an object with known geometry and scale, the object size in the image plane can be used to recover its approximate true distance using

$$d = \frac{h_Y f_y}{h_v} \tag{3.12}$$

where $h_Y$ is the true object height, $h_v$ is the image plane object height, $f_y$ is focal length and $d$ is the mean object depth. The example here only uses the $Y$ axis

projection, but equally applies to $X$. Ambiguity can be introduced by unknown $h_Y$ or $f_y$ and the local geometry must be accounted for.

If an isotropic scaling model is assumed, scale ambiguity can be captured using a single factor $S$ which is applied to a canonical object size $h_{cX,cY,cZ}$ to generate the size of a specific object instance $h_{X,Y,Z}$.

$$\begin{bmatrix} h_X \\ h_Y \\ h_Z \end{bmatrix} = S \begin{bmatrix} h_{cX} \\ h_{cY} \\ h_{cZ} \end{bmatrix} \tag{3.13}$$

### 3.4.2 Method

This study investigates whether a single range measurement can resolve both scale and focal length ambiguity when predicting depth maps from monocular images. This is first tested using two existing datasets, NYUv2 which is indoor RGBD scenes, and KITTI which consists of autonomous driving scenes with stereo RGB and aligned lidar data. These are both modified to simulate variable scales and focal lengths, along with range measurements constructed using the ground truth for each image. Two new datasets are also tested, a set of tabletop objects and one from an outdoor agricultural scene. These are gathered using a custom designed camera cluster which includes an RGBD, wide angle and narrow angle camera. It also mounts a single reading, time of flight (ToF) chip for range measurement, as shown in Figure 3.15.

All four datasets are used to independently train and test the Sparse-to-Dense network architecture presented in Ma and Karaman (2018). This takes an RGB plus range measurement and predicts the full image depth map. For the NYUv2 dataset the ResNet50 backbone is used, while KITTI uses ResNet18. Two sampling techniques are compared, the simplest of these is $n$-Random where $n$ depth values from the ground truth depth map are randomly selected and added to their corresponding positions in the input depth map. Projective sampling applies Equation 3.11 to simulate what reading the camera test rig would measure if applied to that dataset.

This projection approach makes use of the transform between the ToF and D435 RGB axes and uses the intrinsics matrix for this camera. Lidar depth samples form a sparse ground truth in the KITTI dataset so this uses 1-Nearest sampling where the nearest non-zero depth reading to the projective model is used. RGB input forms a baseline where an empty depth map is used at the input. For both newly gathered datasets the ToF sensor reading is used to set a single pixel of the input depth map.

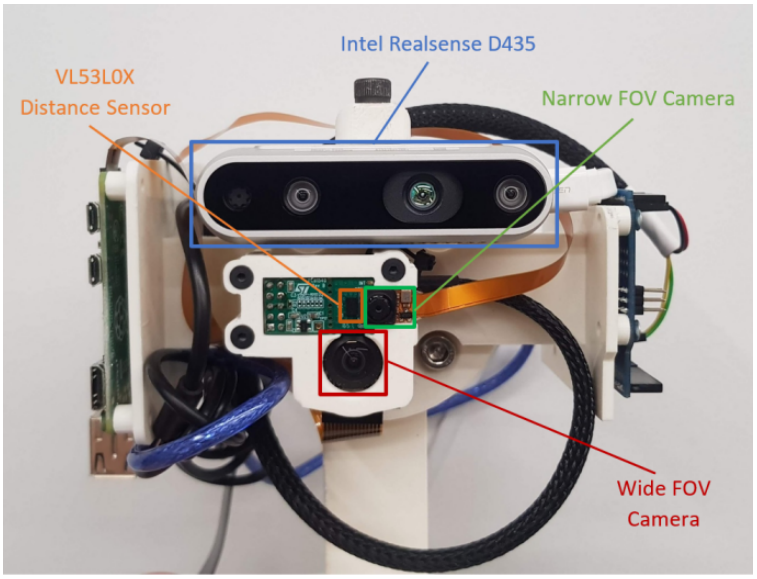


**Figure 3.15** – The cameras and range sensor mounted in the tested configuration

Focal length variation is added using a crop and resize operation on the RGB input, the size of which is randomly sampled from a uniform distribution for each training pair of one RGB image and one depth map. This results in a focal length adjustment of 100% to 150%. To simulate variable scene scales, for each training pair, one of the following values is randomly sampled and the depth map for that image is multiplied by this. Scale set 1 is sampled from $\{0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}$ and set 2 from $\{0.1, 1.0, 1.5\}$.

A tabletop dataset is gathered by imaging 14 common household objects using the camera test rig. This is positioned using a UR5 arm along a set trajectory consisting of 21 views captured over 3 different heights. Each image has a binary object mask manually annotated, to separate the target object and background. Both RGB camera

images, along with the depth map, are aligned with the D435 RGB frame through a rigid image transform to maximise binary mask overlap. Loss on this datasets is only calculated for the object mask pixels, excluding the background.

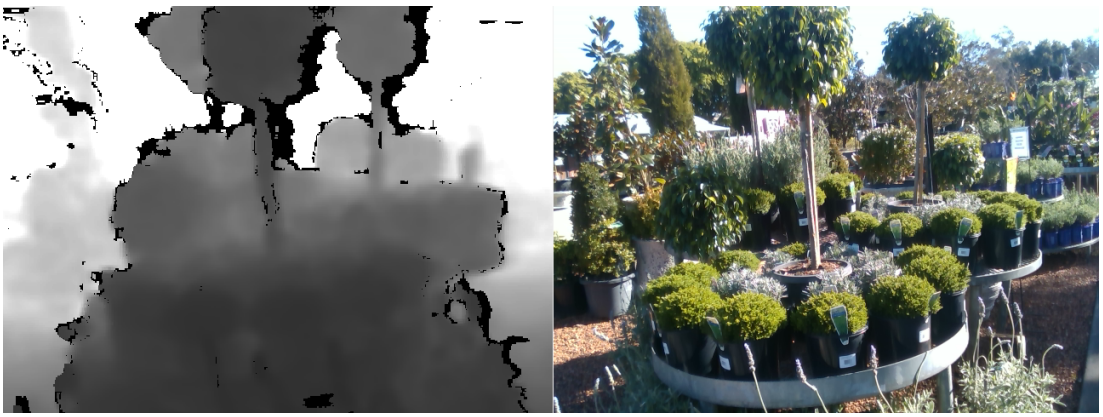

**Figure 3.16** – A sample frame from the outdoor dataset where several similar looking plants of various scales can be seen.

An outdoor dataset is also gathered by imaging a plant nursery which has a wide variety of shapes and sizes of flora. Figure 3.16 shows an example frame with both RGB and depth data. Operating outdoors leads to variable lighting with under and over-exposed frames. Many complex geometries are captured, with object scale ambiguity present in some plant types. Performance of the ToF sensor under direct sunlight was poor and only 4% of the original images had valid depth readings. This resulted in 61 frames for the transfer learning dataset, 40 of which form the training set. Further method and dataset details are available in Brown and Sukkarieh (2019).

### 3.4.3 Results

Experiments are run for various combinations of the projection method, dataset and simulated depth ambiguity conditions. Error metrics of root mean squared (RMS) error in metres, mean absolute relative error and $\delta_1$ from Ma and Karaman (2018) are reported. The second of these is a unitless ratio, and $\delta_1$ is the percentage of pixels with a relative error within 25% of the true value.

### NYUv2

Sampling a single random depth pixel as input was found to be worse than RGB-only in the results of Ma and Karaman (2018) which is also reflected in these results. However, projective sampling of a single point outperforms both of these. Inputting 100 points from random locations significantly improves the performance on all tested configurations.

On the simulated focal length ambiguity data, projective sampling provided no advantage over RGB. Comparison of RMS error values is not informative when either simulated ambiguity type is applied. When multiple scene scales are simulated, projective sampling consistently outperforms RGB-only.

With both forms of ambiguity present, RGB accuracy further degrades, but a single projective sample is capable of producing depth map estimates only slightly worse than when a single form of ambiguity is present.

| **Sampling** | **Variable Focus** | **Variable Scale** | **RMSE** | **Mean Abs Rel** | $\boldsymbol{\delta_1}$ |
|---|---|---|---|---|---|
| RGB | N | N | 0.555 | 0.156 | 79.2 |
| 100-Random | N | N | 0.273 | 0.055 | 96.0 |
| 1-Random | N | N | 0.598 | 0.167 | 75.2 |
| 1-Projective | N | N | 0.546 | 0.150 | 79.8 |
| RGB | Y | N | 0.608 | 0.175 | 74.8 |
| 100-Random | Y | N | 0.262 | 0.051 | 96.5 |
| 1-Projective | Y | N | 0.607 | 0.166 | 76.0 |
| RGB | N | Set 1 | 1.021 | 0.389 | 40.0 |
| 100-Random | N | Set 1 | 0.368 | 0.065 | 95.1 |
| 1-Projective | N | Set 1 | 0.623 | 0.172 | 74.0 |
| RGB | N | Set 2 | 1.652 | 3.438 | 30.3 |
| 100-Random | N | Set 2 | 0.338 | 0.089 | 92.0 |
| 1-Projective | N | Set 2 | 0.588 | 0.179 | 74.0 |
| RGB | Y | Set 2 | 1.927 | 4.136 | 16.4 |
| 100-Random | Y | Set 2 | 0.202 | 0.074 | 93.5 |
| 1-Projective | Y | Set 2 | 0.456 | 0.187 | 72.8 |

**Table 3.4** – Depth map prediction performance using monocular imagery from the NYUv2 dataset for various sampling methods and scale sets.

### KITTI

When using single point sampling, the accuracy was less than just RGB input for the KITTI dataset. KITTI does not have dense ground truth depth data, so the nearest available depth pixel to the projective pixel location is used. This additional variance in the range measurement pixel location, compared to NYUv2, may explain the worse performance. Dataset differences may also contribute, a randomly selected pixel from KITTI is more likely to fall on the background than a distinct object, as compared to NYUv2. KITTI is expected to have lower object scale ambiguity as road scenes feature well defined object classes of similar scales. Comparable results to NYUv2 were otherwise seen. A single range measurement outperforms RGB-only when variable scene scales are simulated but focal length ambiguity alone cannot be distinguished by projective sampling.

| **Sampling** | **Variable Focus** | **Variable Scale** | **RMSE** | **Mean Abs Rel** | $\boldsymbol{\delta_1}$ |
|---|---|---|---|---|---|
| RGB | N | N | 4.704 | 0.116 | 85.5 |
| 1-Nearest | N | N | 4.834 | 0.117 | 84.1 |
| RGB | Y | N | 4.528 | 0.107 | 87.3 |
| 1-Nearest | Y | N | 4.640 | 0.110 | 86.5 |
| RGB | N | Set 2 | 11.553 | 2.983 | 30.4 |
| 1-Nearest | N | Set 2 | 7.124 | 1.127 | 64.1 |
| RGB | Y | Set 2 | 11.822 | 3.098 | 31.1 |
| 1-Nearest | Y | Set 2 | 7.505 | 1.165 | 65.9 |

**Table 3.5** – Depth map prediction performance using monocular imagery from the KITTI dataset using two sampling methods and one scale set.

### Experimental Test Datasets

Depth map prediction performance on the new indoor and outdoor datasets was poor, as summarised in Table 3.6. Despite applying transfer learning, in the form of imagenet pretrained weights, to minimise data requirements, the network was unable to predict accurate depth maps. This is likely caused by changes in the scene

appearance and significantly different average scene scale to NYUv2 and the small training dataset set size.

| Sampling | RGB Source | RMSE | Mean Abs Rel | $\delta_1$ |
|---|---|---|---|---|
| Bench Top Dataset | | | | |
| RGB | Wide | 0.565 | 0.190 | 75.6 |
| 1-TOF | Wide | 0.668 | 0.227 | 60.1 |
| RGB | Standard | 0.449 | 0.164 | 79.1 |
| 1-TOF | Standard | 0.528 | 0.192 | 67.5 |
| RGB | Narrow | 0.664 | 0.208 | 65.8 |
| 1-TOF | Narrow | 0.792 | 0.306 | 55.7 |
| Outdoor Dataset | | | | |
| RGB | Standard | 2.719 | 0.398 | 38.6 |
| 1-TOF | Standard | 3.05 | 0.476 | 29.6 |

**Table 3.6** – Depth map prediction performance using monocular imagery of indoor and outdoor scenes generated using the experimental testing hardware.

### 3.4.4 Monocular Depth Inference Study Conclusion

Results in Table 3.4 indicate that a single range measurement can resolve large scene scale ambiguity in practice. However, the performance of this technique on focal length changes and datasets without large amounts of scale ambiguity, such as KITTI, is mixed. Projective sampling is important, and using multiple input points was found to be very effective but requires significantly larger and more expensive hardware to do this.

Serious limitations in the ToF sensor chosen were observed when the outdoor dataset was gathered. A 2m range was insufficient for many frames, while changing light conditions in direct sunlight caused issues with ground truth depth maps and ToF readings with the hardware used. Transfer learning from NYUv2 to the new outdoor dataset was not effective and additional training data is likely needed to apply this technique in such environments. Gathering sufficient amounts of training data is not entirely straightforward. The shadowing, occlusion, alignment error, and sparseness

of depth maps produced by the D435 causes artefacts that may be learned by the network. Alternate depth camera models may perform better outdoors and should be explored for this application. Impacts of depth ambiguities on current monocular depth prediction approaches have still not been fully explored. The fractal structure of plants means that depth ambiguity may be a more significant problem in agricultural scenes than in built environments.

Applying this technique to the harvesting prototype would require an additional ToF sensor, reader board and the aforementioned large training datasets to be effective. Even with these in place, predicting constantly variable organic scenes is more difficult than the NYUv2 case, and performance on that is insufficient for fine manipulator control. For these reasons, monocular depth inference is not applied to the prototype harvesting system.

# Chapter 4

# Fruit Localisation

With the environmental representation of geometric primitives selected and the sensing approach chosen, the next functional module consists of fruit localisation. This occurs first in the 2D camera frame, the 3D pose is then extracted using depth information and filtered over time. This approach leverages the high performance of object detectors for 2D imagery, and the available depth information. Section 4.5 explores an alternative method for localisation.

As illustrated in Figure 4.1, object detection is first used to place a bounding box around the extents of each fruit in a 2D image. Once an image patch containing the fruit is known, a highly tolerant hue-saturation-value (HSV) filter is applied to segment pixel locations on the actual fruit. This excludes leaves and stems, which are not solid regions for grasping. These same pixel locations are extracted from the depth map and the target depth estimated as their median. Fruit depth is combined with the pinhole camera projective model to determine the world frame fruit position. Estimated positions from multiple frames are finally combined into a single filtered estimate which tracks the location of each target fruit. These filtered locations are passed to the grasping modules described in Chapter 5.

Three specific areas of the localisation process are chosen for in depth studies. A comparison of multiple architectures for fruit detection in harvesting identifies the best of these, reducing false detections and missed fruit under both day and night condi-

tions. An autocovariance least squares method for tuning noise covariance matrices in PBVS is presented, reducing filtering error and manual tuning effort. An EKF based active perception approach that can function without camera depth sensing is explored, which could reduce system errors when depth is unavailable.

## 4.1 Object Detection

Accurately and consistently identifying fruit in 2D camera images is a key requirement for effective fruit localisation in this system. Like most current research efforts, this is approached with deep learning tools which require large training datasets. Unfortunately, when working with seasonal produce, images of the target fruit captured under harvesting conditions may not be available prior to initial field trials. Instead a two phase approach was used with the prototype.

An initial detection system is developed to meet the engineering goals of the platform, such as framerate, training time and compute requirements. This is used for detection and harvesting during field trials. Data gathered from these trials is then applied to perform a detector performance comparison study, to select the best architecture for future trials.

An HSV thresholding detector was first tested, this technique has been applied to harvesting systems in the past and is extremely simple. While HSV performed adequately for pre-testing on apples, the Victorian plum crop could not be well segmented using this approach due to similar soil colouring. Initial testing in the field immediately ruled out using this method, which was confirmed by more rigorous lab evaluation. Using the data from Section 4.2 a recall rate of 25.1% and precision of 80.3% was seen for the HSV filter, with 42.1% of the correct detections having an inaccurate bounding box.

Yolov3 is selected as the initial deep network with the default parameters used. This is easy to train and deploy on the embedded inference computer described in Section 6.1.2, with competitive accuracy and speed among current generation detectors.

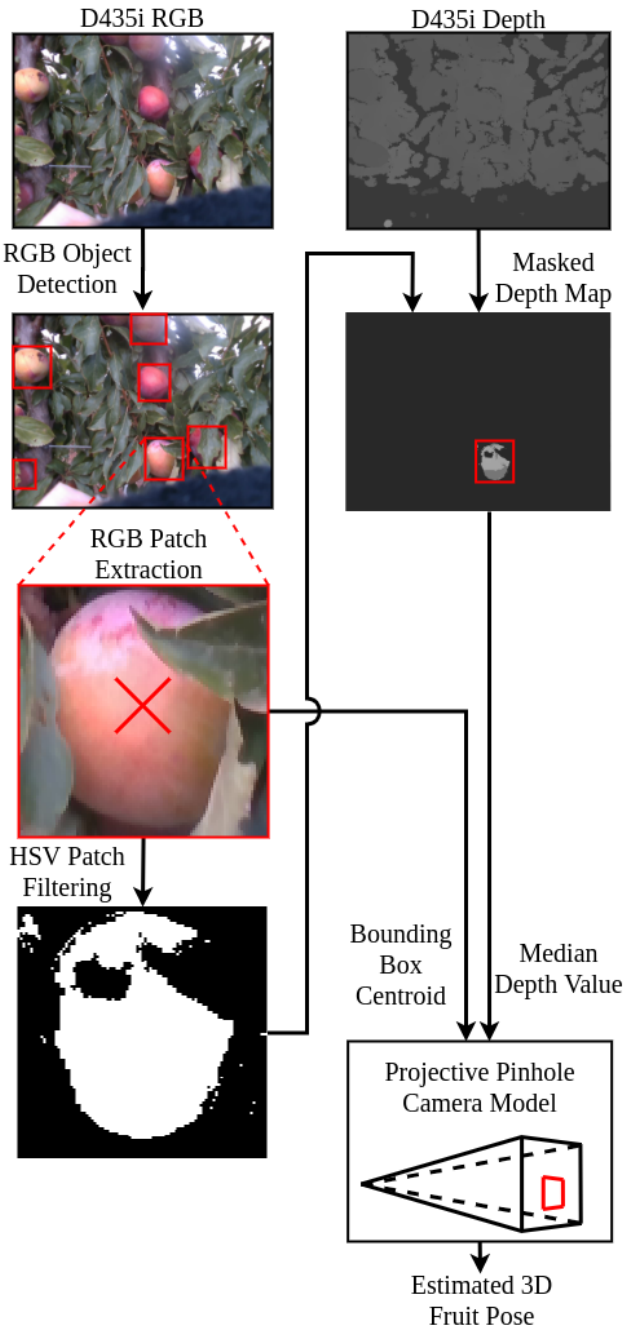


**Figure 4.1** – Fruit localisation uses the D435i RGBD data. First, object detection is run on RGB images, followed by HSV filtering on each bounding box patch. The HSV mask is used to select which depth map pixels are used for median depth calculation. The bounding box centroid and median depth, corrected for fruit diameter, are applied to the pinhole projective model to determine target pose, which is passed to the EKF.

Maximising in the field training effectiveness was a key goal, so Yolov3 is pre-trained on plum images scraped from the internet.

When deployed to the field trial, 100 images of approximately 700 plums, were gathered on the first day and Yolov3 was fine tuned overnight. Better detector performance yielded more harvesting attempts on the following day, giving more training data for the detector fine tuning. This cycle was repeated a total of 3 times, over 3 days, eventually resulting in an effective object detector by the trial conclusion. This field trial is further described in Chapter 6. Addressing retraining amid seasonal variations is one complexity of commercial autonomous harvesting. The experiences here suggest overnight transfer learning is feasible and only a few days of harvest time are lost to object detector training for each new cultivar variety. Returning to the same fruit in the future should reduce this timeline further, but training should begin several days before the first harvest is due to minimise delays.

For the embedded Yolov3 model, recall was 32.2%, with a precision of 100%, and only 2.7% of bounding boxes were inaccurate. This is significantly better than HSV detection but still lower than expected. The poor detector performance motivated a standalone comparative study to address the challenges of object detection for harvesting.

Eye-in-hand sensing provides distinct advantages for fruit harvesting by allowing for continuous feedback control right up to the point of picking. This decreases positioning error upon gripper final approach. However, as the hand and camera move from viewing the entire trellis to picking a specific fruit, large changes in lighting, colour, obscuration, and exposure occur. These issues can be seen in Figure 4.2 and make detection for eye-in-hand harvesting a uniquely challenging problem.

Controlled lighting is one technique for dealing with illumination changes, but introduces extra complexity and cost. It is challenging in daylight conditions and must be co-located with the eye-in-hand camera to avoid being obscured. Having object detectors tolerant to most illumination issues is a more convenient solution if possible.

Obscuration is a fundamental part of crops that grow within a tree canopy and plums

**Figure 4.2** – Clockwise from top left are examples of; a typical night time eye-off-hand viewpoint, a typical day time eye-in-hand viewpoint, blurring due to low light levels at night, and exposure and colour changes due to the camera entering the sun after being obscured by leaves.

which are not at all obscured, often suffer from sunburn, reducing their quality. Using multiple views in a filtering and tracking framework allows fruit to be harvested even if they can only be seen from some camera viewpoints. Detectors capable of identifying mostly obscured fruit are important for this to function properly.

## 4.2 Study: Object Detector Comparison

While Yolov3 was chosen for the initial harvesting trial, this choice was partially based on familiarity and ease of implementation. Using the images gathered during the trial, various modern object detection architectures can then be properly compared for their

performance on this task. In this study, 4 deep learning object detection networks are assessed against day and night datasets gathered during the prototype harvester field trials. Additionally, two methods for fusing depth and image information are tested for their impact on detector performance. Significant differences between day and night accuracy of different detectors were found, transfer learning is identified as essential in all cases, and depth information fusion is assessed as only marginally effective.

### 4.2.1 Method

To test the performance of current generation object detector architectures for eye-in-hand harvesting, images were gathered from the D435i RGBD camera during the prototype field trial. A total of 700 images were extracted and annotated, these are split evenly between day and night datasets. Two previous generation object detection deep learning networks, Faster-RCNN and YoloV3, are benchmarked on this dataset, along with two current networks; RetinaNet and CenterNet. Each network architecture is trained and tested on the day and night datasets separately. Both pretrained weights for transfer learning from a non-agricultural task, and randomly initialised weights are used with each architecture.

Because depth is required for fruit localisation, the fusion of RGB and depth information for detection is also tested on the day and night datasets. Both early and late data fusion is trialled using the RetinaNet architecture. Early fusion treats the network input as a 4 dimensional RGBD image, while in late fusion the image and depth features are concatenated after being extracted by parallel network backbones.

#### Dataset

During the field trial described in Chapter 6, image feeds from the Realsense D435i were recorded. An embedded version of the YoloV3 model trained on previous data was used to detect harvesting targets with the D435i, all detected fruit were attempted, resulting in many failed picks in the current dataset. Several hours of

sensor data were recorded during harvesting over the period of one day with direct sunlight, shadowed sunlight, and overcast conditions. For night time operation, a single diffused floodlight was used, mounted off the arm.

During harvesting, the image frames are processed at a rate of 10.5 per second. The camera driver provides on-board depth map alignment to the RGB images and all data is at a resolution of 640x480px. Images for the day dataset were extracted from the camera feed at 0.5 second intervals, then 350 were manually selected to form a representative dataset. Frames that did not contain plums, were excessively blurry, or similar to existing frames were not selected. Colour balance, distance to the trellis, and number of targets were not considered when selecting frames. This process was repeated for data gathered at night. These two datasets of 350 images each are then manually labelled with bounding boxes around all visible plums and split into train, test and validation subsets of 176, 87, and 87 images respectively. A total of 4449 plums are annotated in the day dataset, and 1402 in the night. Fewer plums were seen at night due to the light source not fully penetrating the canopy.

Between pick attempts, the camera is positioned approximately 70cm from the trellis for a global view of the trellis area being picked. Because the target position estimates are continuously updated from all sensor frames, the object detector must be robust to both near and far viewpoints of fruit, as seen in the dataset.

Many of the gathered images exhibit numerous artefacts directly related to the harvesting task. Some of these are caused by the camera motion as it moves from the global pose to harvesting a fruit. This results in a wide range of distances, bounding box sizes and illumination changes, as seen in Figures 4.3 and 4.4. Most images also include part of the gripper, a design trade-off necessary to minimise the camera and gripper footprint. Exposure and white balance are handled automatically by the camera driver and must be variable to deal with changing light conditions throughout the day. Occasionally, this results in extremely mis-exposed or mis-coloured images which the system should be tolerant to and are present in the datasets in small numbers.

Depth imagery is required for localising the fruit after they have been detected. Because this sensor modality is already present and is increasingly common among

**Figure 4.3** – Example RGB, RGBD overlay and depth images from the day dataset. Including effects specific to harvesting motion such as large illumination, object size and obscuration changes. Depth images shown after clipping is applied.

automated agricultural platforms, the fusion of RGB and depth data was tested. To create the RGBD dataset, each annotated RGB image has a corresponding depth image included as a separate file. Depth data suffers from holes where the range is out of sensor limits, shadowing where a point is obscured for one of the IR stereo pair used to calculate depth, and also from smoothing effects in the depth calculation algorithms. Various methods have been proposed to overcome these limitations, however, for simplicity, the depth data is only normalised and clipped before being used in network training. Clipping occurs by setting values below 0.11m, where the camera can return incorrect readings, to be zero and values above 2.5m to be 2.5m. All depth readings are then divided by 2.5m to produce a pixel range from zero to one. At short ranges, many depth errors are still present, seen in Figure 4.3 column 3.

The datasets are formatted to match the Pascal visual object classes (VOC) 2007 standard from Everingham et al. (2010). The two RGBD datasets and trained models

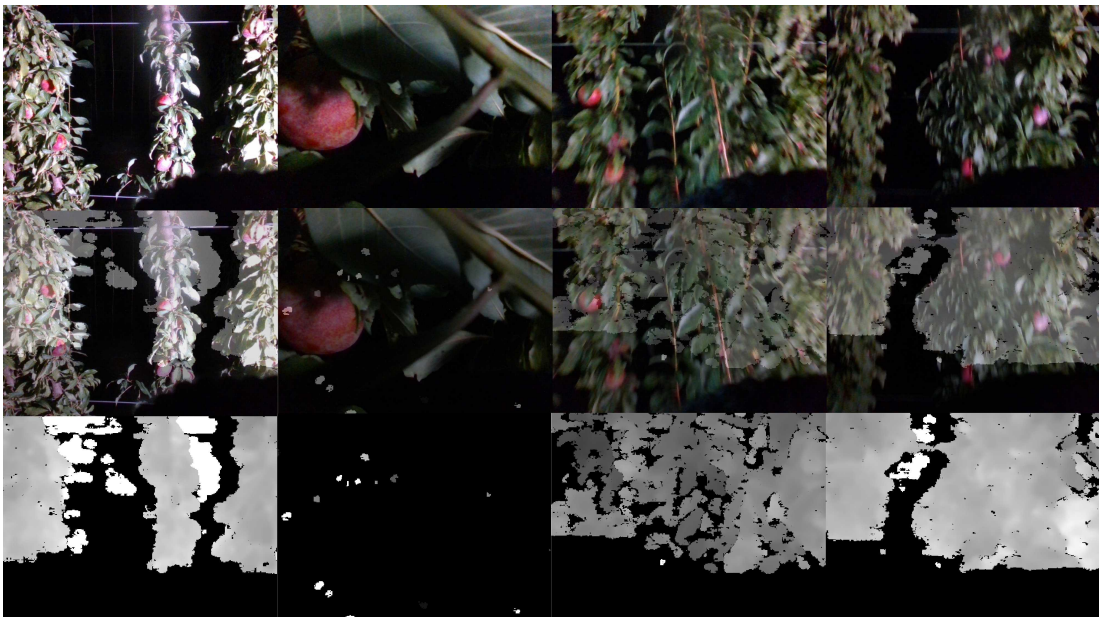

**Figure 4.4** – Example RGB, RGBD overlay and depth images from the night dataset. Depth performance is improved at night with better defined object edges and less smoothing effects. Depth images shown after clipping is applied.

for these are made available online[8].

### RGB Network Architectures

Four commonly used object detector networks were chosen for evaluation; Faster-RCNN, YoloV3, RetinaNet and CenterNet. The first three use Keras implementations, while CenterNet is in PyTorch. These span a range of target frame rates and all lie on, or close to, the outer edge of the speed-accuracy curve for the standard computer vision dataset common objects in context (COCO) from Lin et al. (2015). This can be seen in Table 4.1.

Faster-RCNN, from Ren et al. (2017), is the only two-stage detector tested, an approach which shows improved accuracy over single stage detectors, at the cost of slower inference times. YoloV3 is a single-stage detector used in many existing works looking at object detection for agriculture, and remains a competitive detector for high

[8]http://data.acfr.usyd.edu.au/Agriculture/PlumDetection/

frame rate applications. Updated versions of Yolo are also available, see Bochkovskiy et al. (2020). The Yolov3 version tested here is separate to the original C embedded implementation used during the field trial.

Lin et al. (2020) present RetinaNet. This implements the concept of focal loss which alters the loss function to downweight the impact of easy negative examples, where there are clearly no objects within the bounding box. CenterNet, by Duan et al. (2019), is the newest and largest network tested. A one-stage approach is used to predict heat maps of where bounding box corner and center points lie.

| **Network** | **Backbone** | **Input Size (pix)** | **COCO APM** | **Inference Time (ms)** |
|---|---|---|---|---|
| Faster-RCNN | VGG-16 | 600xN | 34.7 | 250 |
| YoloV3 | DarkNet-53 | 416x416 | 33.0 | 29 |
| RetinaNet | ResNeXt-101-FPN | 800xN | 40.8 | 198 |
| CenterNet | Hourglass-104 | 511x511 | 47.0 | 340 |

**Table 4.1** – The originally published COCO average precision metric for each object detection architecture. Additionally, the reported model inference time, although each model uses different GPU hardware, so these are only roughly comparable. Faster-RCNN figures are from Huang et al. (2017) who note that model speed is highly dependent on the number of box proposals. Faster-RCNN and RetinaNet resize the input to make the short image edge match the stated value.

All of the precise training configurations applied to these networks during benchmark testing can also be found at the dataset web page.

### RGBD Network Architecture

Two forms of information fusion, early and late, are commonly presented in the literature. Both are tested here using the RetinaNet architecture against an RGB-only baseline.

Early fusion refers to concatenating the depth information as an additional input channel, in our case this makes the network input a $480 \times 640 \times 4$ tensor, prior to resizing. Early fusion is easily implemented and does not significantly increase computational requirements, but shows mixed results in the literature, often performing worse than RGB alone. Figure 4.5 shows the early fusion network.

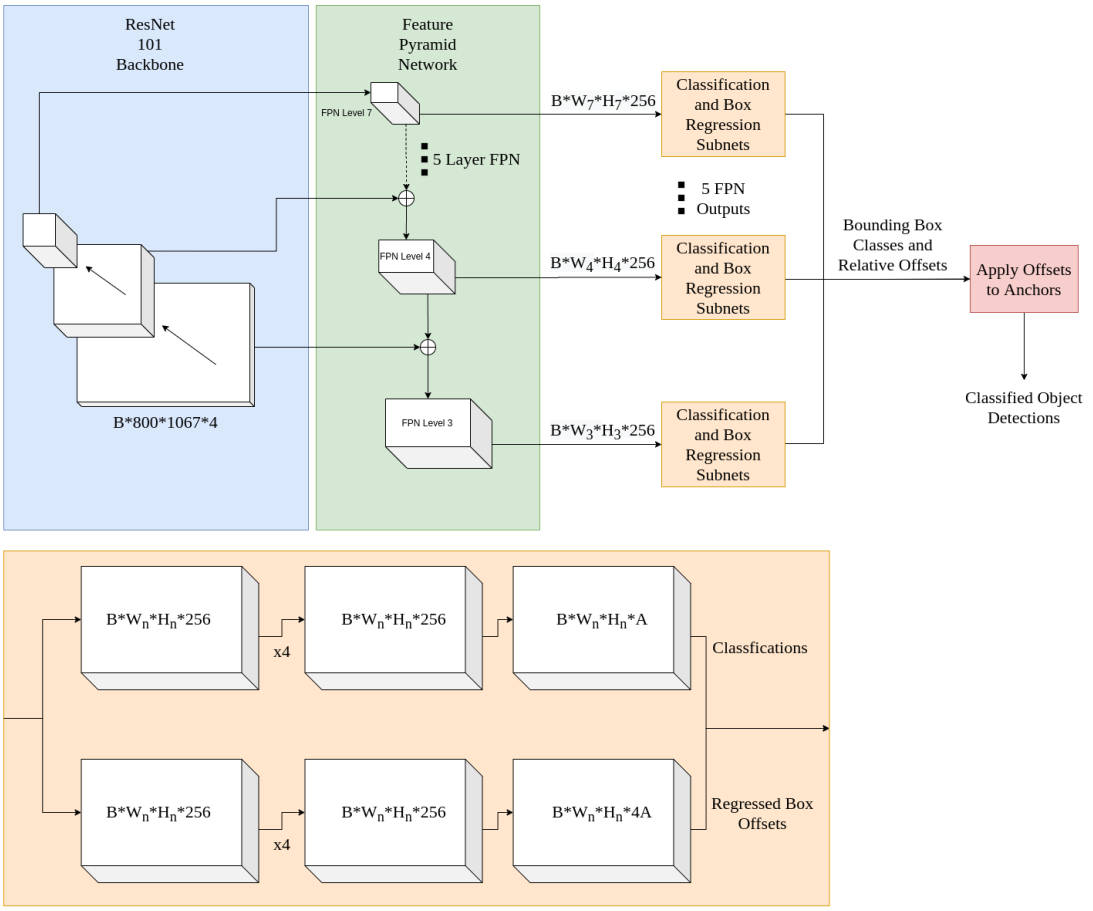


**Figure 4.5** – The early RGBD fusion network using the ResNet-101 backbone, adapted from Lin et al. (2020) and identical to their implementation apart from the input layer shape. B is the batch size, A is the number of anchors, there are a total of 5 feature pyramid network (FPN) levels used which are numbered 3 to 7 to match the above mentioned paper.

Late fusion runs a pair of feature extractor backbones and FPNs, on the RGB and depth data in parallel. At each of the 5 FPN scales, features from the RGB and depth FPN outputs are channel wise stacked before being passed through a 1x1 convolution which performs pooling over the RGB and depth feature maps. This reduces the FPN channels to 256 so the classification and regression subnetworks are identical to the RGB-only case. The overall network size is slightly less than doubled. Addition of an extra backbone creates more informative features which can be learned specifically for the depth modality, at the cost of additional complexity and execution time. Depth

data has a meaningful absolute value and is pre-normalised to a fixed range, so batch normalisation layers are removed from the depth backbone.

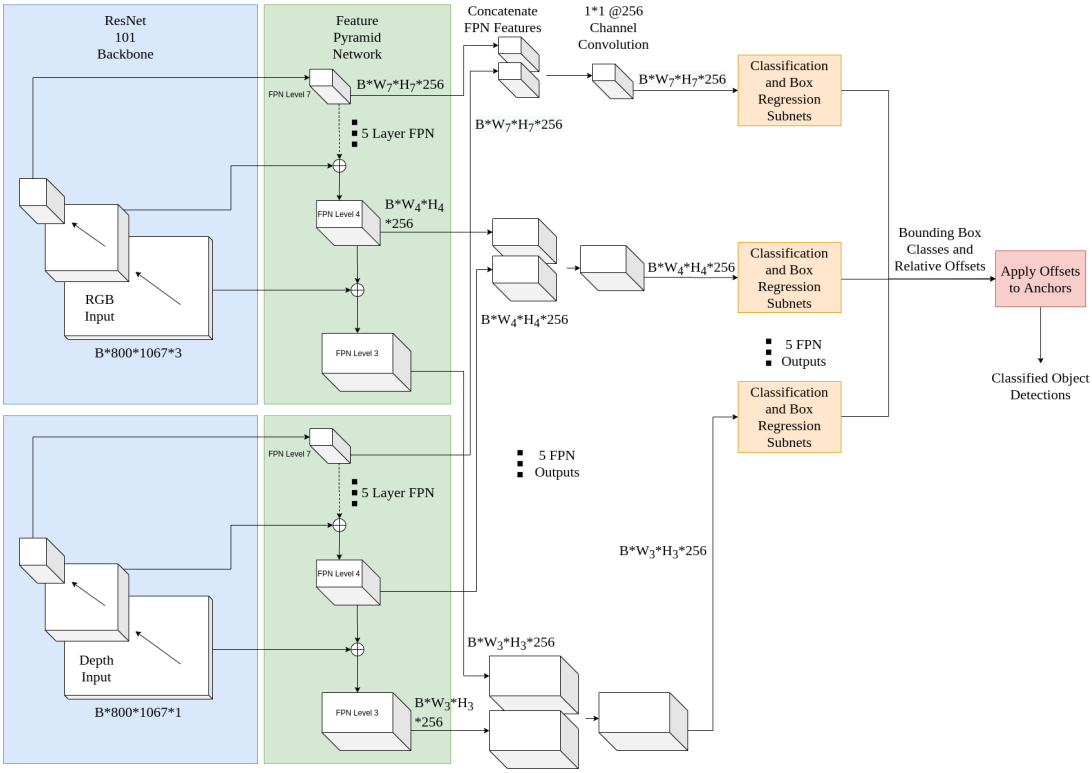


**Figure 4.6** – The late RGBD fusion network, adapted from Lin et al. (2020). The backbone and FPN is duplicated, output features from this are channel wise stacked and passed through a $1 \times 1$ convolution to reduce the channels back to 256. The same two subnets as Figure 4.5 are then applied.

### Training & Testing Details

All object detection networks can be improved through careful hyperparameter tuning. To provide a fair comparison, and because many application areas lack the resources for extensive network tuning, each architecture is trained using the default parameters provided by the authors. For YoloV3 and CenterNet, these were found using the COCO dataset, while Faster-RCNN and RetinaNet were primarily developed using Pascal VOC.

Modifying the default anchor sizes to match the mean plum size for the dataset, with

anchor proposals both larger and smaller than the mean, was found to be counterproductive for all networks. So default anchor sizes are used. All architectures do some form of data augmentation by default, specifics of which can be found at the public link provided. Networks are trained until the validation loss plateaus. All inference time results were achieved using an Nvidia GTX 1080Ti and the training batch sizes are set to the maximum that can fit on this GPU.

Transfer learning refers to using weights from an already trained model as the starting point for training on the day and night plum datasets. All networks are tested both with and without transfer learning.

All results are processed using the official VOC2007 Matlab development kit. Evaluation is done by plotting the precision-recall (PR) curve and reporting the average precision metric (APM) of Everingham et al. (2010), with a bounding box intersection over union (IOU) threshold of 0.5.

Depth data is absolute in nature and relating scene geometry to image data requires the camera focal length. So, to preserve correlated features between RGB and depth inputs, a fixed focal length should be used. This prevents the use of image re-scaling and the augmentations, such as cropping, translation and rotation, which rely on it.

For all six RGBD tests, image augmentation is disabled and transfer learning is applied using ImageNet weights. The dual backbones were found to make late fusion training unstable so a two step process is required to effectively train this network. First the depth backbone is frozen and the RGB ResNet, FPN and subnet modules are trained, then all layers are then unfrozen and the depth backbone is also trained.

### 4.2.2 Results

Each RGB network is tested against the day and night dataset separately, both with and without transfer learning. The impact of depth fusion is assessed using the RetinaNet architecture. The RGBD fusion tests, including the RGB baseline, do not use data augmentation, whereas all other networks do. All tested configurations are summarised in Table 4.2 with PR curves for each dataset shown in Figure 4.7.

Some training runs were unstable, resulting in no validation set APM increase during training. Each unstable training configuration was tested three times and in all cases the three runs failed. No training runs failed where an APM value is reported.

| Architecture | Backbone | Configuration | Day APM @0.5 IOU | Night APM @0.5 IOU | Mean Inference Time (ms) |
|---|---|---|---|---|---|
| Faster-RCNN | VGG-16 | Transfer Learned | 0.691 | **0.795** | 128 |
| | VGG-16 | Random Weights | 0.537 | 0.788 | |
| YoloV3 | DarkNet-53 | Transfer Learned | 0.597 | 0.746 | **56** |
| | DarkNet-53 | Random Weights | Unstable | 0.608 | |
| RetinaNet | ResNet-50 | Transfer Learned | 0.781 | 0.778 | 72 |
| | ResNet-50 | Random Weights | 0.639 | 0.744 | |
| RetinaNet | ResNet-101 | Transfer Learned | **0.787** | 0.767 | 102 |
| | ResNet-101 | Random Weights | Unstable | Unstable | |
| CenterNet | Hourglass-104 | Transfer Learned | 0.709 | 0.746 | 276 |
| | Hourglass-104 | Random Weights | 0.456 | 0.632 | |
| Retinanet RGBD | ResNet-101 | Early Depth Fusion | 0.608 | 0.732 | 109 |
| | ResNet-101 | Late Depth Fusion | **0.745** | **0.781** | 143 |
| | ResNet-101 | RGB Baseline | 0.730 | **0.781** | **99** |

**Table 4.2** – Results for each object detection network on the day and night datasets. APM is calculated using the VOC2007 development kit, mean inference time is per image, not including network loading time.

### RGB Only

Over the four baseline networks tested, RetinaNet with ResNet-101 achieved the highest APM on the day dataset while Faster-RCNN performed the best on the night data. Transfer learned networks were much more accurate than those trained from scratch, while also taking less time to train.

YoloV3 was the fastest network by a significant margin, although with lower than average accuracy. RetinaNet with ResNet-50 provides a good speed-accuracy trade off for most applications. Data augmentation using the RetinaNet default methods was effective, as shown by the difference between the RGB baseline from the RGBD tests and the ResNet-101 transfer learned results.

Faster-RCNN, YoloV3, and CenterNet all performed better on the night dataset.

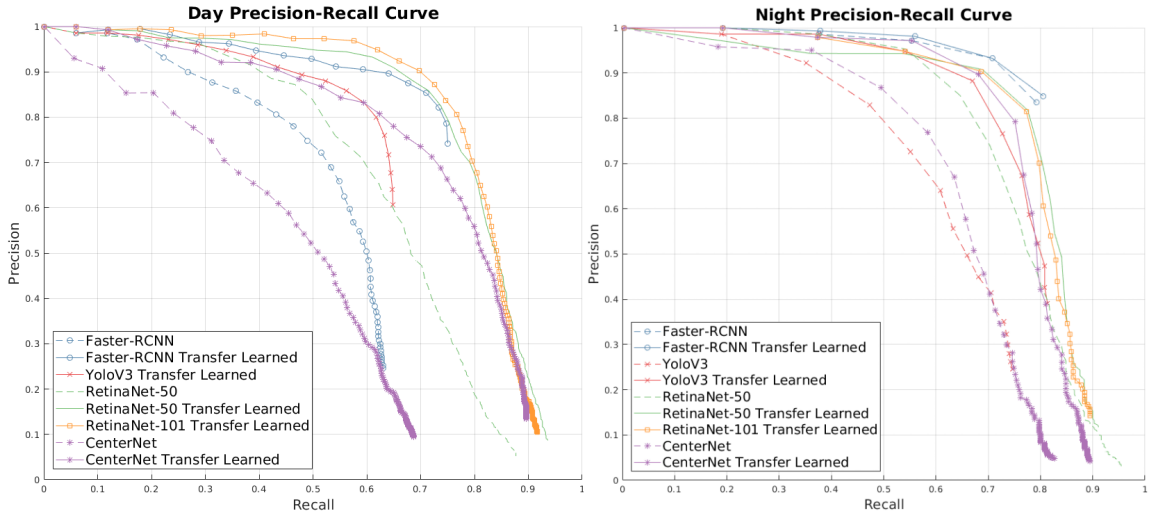


**Figure 4.7** – The PR curve for each architecture on the day time dataset (left) and night time dataset (right).

Fixed lighting conditions and fewer visible but obscured fruit should make this an easier detection task, although there are less training instances available.

#### RGBD Fusion

Precision-recall curves for the RGBD tests are plotted in Figure 4.7. Early fusion performed worse than RGB alone, even with other factors such as data augmentation, being equal. Late fusion slightly outperformed the baseline on both day and night data. Doubling of the network backbone produced only a 31% increase in inference time. Many operations such as image pre-processing and bounding box non-maxima suppression are not dependent on network size.

### 4.2.3 Discussion

Differences between performance reported on the COCO dataset, and the two datasets tested, were surprising. CenterNet performed poorly on both plums tasks, despite having the highest stated COCO accuracy, it was also the slowest. RetinaNet was effective for day time detection, with augmentation, transfer learning and backbone

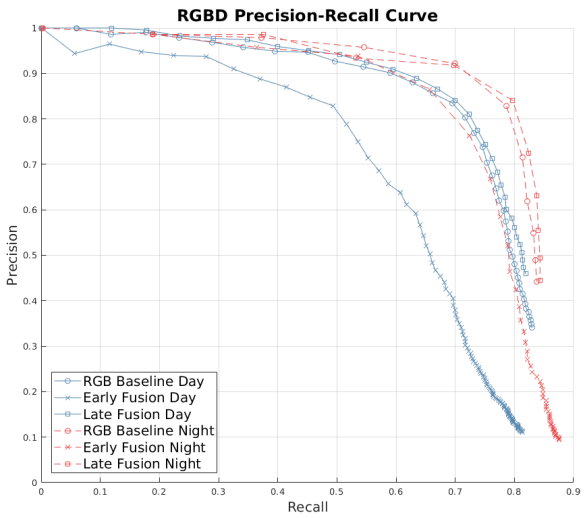


**Figure 4.8** – The PR curve for the three RGBD fusion approaches on both datasets.

size all playing a role in overall performance. Direct comparison to existing works is not possible, owing to the lack of standardised datasets for fruit detection during harvesting.

Applying transfer learning had an overall larger impact than architecture selection and is essential when using small datasets, as in many agricultural applications. Conversely, using the ResNet-101 backbone significantly increased RetinaNet processing time, for only a small benefit in precision. Design decisions such as these can play a more essential role than architecture choice, and should be carefully considered. Additional factors, such as dataset and batch size, are not investigated in this work but typically also have an impact on accuracy.

Faster-RCNN outperformed both more modern and slower networks on the night dataset, in contrast with the daytime performance. Fortuitous hyperparameter defaults may be a contributing factor, though properly exploring these for all 10 RGB configurations is not feasible. This result highlights the importance of testing a range of network types on application-specific data, such as harvesting under controlled lighting. High intensity strobe lighting should be trialled to reduce variability during the day and night time blurring, the floodlight tested was not bright enough to be seen in images during daylight.

Early data fusion was counterproductive for these datasets, and performed worse than RGB alone, as shown in Table 4.2. Although late data fusion was effective, the gains from this method were less than that provided by data augmentation, and expanding the RGB dataset size would likely be significantly more useful than incorporating depth information. Additionally, longer ResNet backbones can be constructed, and typically show a small APM improvement over ResNet101. So the additional network capacity introduced by the late fusion approach may be better used as a longer RGB only backbone.

Predicting the full extent of partially obscured fruit is important for accurately estimating the fruit centroid and directing the gripper to this. All of the tested networks were able to accomplish this, based on the 0.5 IOU threshold and visual inspection of the results. However, no accuracy metric is ideal for all use cases, and the 0.5 IOU threshold is an arbitrary assessment point commonly used in computer vision. Other metrics may be more suitable for tasks such as harvesting, where the IOU threshold required for a successful pick can often be estimated.

Selecting the ideal precision-recall point for harvesting is a complex question that remains to be answered. Low recall causes fruit to be missed, while low precision may cause pick attempts on false positives. Future hyperparameter tuning and architecture optimisation should increase network performance on this task, with the results here acting as a benchmark and dataset for comparison.

### 4.2.4 Detector Comparison Study Conclusion

In this study, two datasets gathered during a robotic harvesting trial on 2D trellis plums were presented, and four deep learning object detection architectures were benchmarked on these. The fusion of depth information was trialled and found to be marginally effective for late fusion, although data augmentation provides a larger performance boost. On the day time dataset RetinaNet was the most accurate, while Faster-RCNN showed the best average precision for the night time data.

Relative network performance differed significantly to that published for the COCO

dataset, which is commonly used when making design decisions for applications in agriculture. Therefore the public availability of a wide variety of application-specific datasets, such as tree fruit harvesting, is important to future progress.

Which detector to use for future harvesting prototypes cannot be definitively chosen from this study. The best option is likely to be fruit and growing condition dependent, without a clear link between established performance on non-agriculture datasets and harvesting detections. However, the transfer learned versions of both Faster-RCNN and RetinaNet performed well on the day and night datasets. In future, extensive parameter tuning of these networks should be explored, along with architecture refinements to make them better suited to plum detection for harvesting.

The primary limitation of this work is the relatively small dataset size, which makes it impossible to test which benchmark networks perform best on very large datasets. Numerous works have shown a clear correlation between training set size and network performance for fruit detection, although the exact performance-by-size function is not linear and varies between architectures. Continuation of this work should involve additional field trials with more opportunities for object detector data gathering. Future investigations into accuracy metrics specific to tree crop harvesting, multi-view detection and multi-spectral imaging would also be beneficial.

## 4.3 Target Tracking

Once fruit are detected in image frames, pose estimation is run on these to yield 3D target positions. Each position estimate is then filtered and tracked using an Extended Kalman Filter. An additional step to remove false detections is then applied. The presence of depth information and already known camera pose, allows simplified target association using nearest euclidean distance, rather than image reprojection distance. This also motivates the use of an EKF for feature space tracking, although other filter types are well suited for image association and tracking, such as the Kanade–Lucas–Tomasi formulation.

Each object detection consists of an RGB bounding box with centroid $(u, v)$, confidence score and image-aligned depth map. Pose estimation first removes detections below a set confidence, then estimates the depth value of the target centroid for each fruit of $n$, then uses this to project it into a 3D pose $x_n, y_n, z_n$. Each pose is tracked within the EKF state vector which is updated by each new frame, resulting in continuous tracking which updates at an average rate of 10.5Hz. When a fruit is selected for harvesting, the most recent state estimate is read from the EKF and used for motion planning. Target motion was not found to be significant during the field trial, so no motion prediction is required. This is summarised in Algorithm 4.1 where $\hat{y}$ is the EKF state which tracks each fruit position, $\Sigma$ is the target state covariance with initialisation value $\Sigma_0$ and $\text{RGB}_{BoundingBoxRegion}$ denotes the RGB image patch within the bounding box extents.

**Algorithm 4.1:** Target Localisation

**Input:** RGBD stream
**Output:** $\hat{y}$
$\hat{y} = \emptyset$;
$\Sigma = \emptyset$;
**while** *new RGBD frame* **do**
  (bounding box, confidence) ← Yolov3(RGB);
  **for** *bounding box* **where** *confidence > threshold* **do**
    image patch ← $\text{RGB}_{BoundingBoxRegion}$;
    binary fruit patch ← HSV(image patch);
    $d = median(\text{DepthMap}_{BinaryFruitPatch})$ + fruit radius;
    `/* If the depth is valid */`
    **if** *camera velocity < threshold* **and** $d < max_d$ **and** $d > min_d$ **then**
      point ← deproject(bounding box, $d$, camera pose);
      observation ← associateToTarget(point,$\hat{y}$);
      **if** *no association* **then**
        $\hat{y} \leftarrow \hat{y} \cup$ point;
        $\Sigma \leftarrow \Sigma \cup \Sigma_0$;
      **else**
        $\hat{y}, \Sigma \leftarrow$ EKFUpdate($observation, \hat{y}, \Sigma$);
      $\hat{y}, \Sigma \leftarrow$ RemoveFalsePositives($X, \hat{y}, \Sigma$);

### 4.3.1 Pose Estimation

Obscured fruit are a problem for this system when naively extracting the depth value for a bounding box. Even with accurate box extents, a spherical fruit will only take up $\frac{\pi}{4}$ of the depth map for those pixels. This means using the mean or modal depth value for the entire bounding box is unlikely to work for even slightly obscured fruit. To further restrict which pixels are considered when estimating fruit depth, a highly tolerant HSV filter is applied to the RGB region within the bounding box. This combines the robustness of deep learning detections with the speed and simplicity of HSV thresholding. In practice, calculating depth as the median of HSV thresholded depth map pixels proved to be much more effective than using the full bounding box. For cases where the HSV filtering fails, the bounding box centroid point depth is used.

To increase processing speed, the RGBD input is downscaled from 640x480 to 320x240. Deprojection of the point occurs in the world frame, so requires parsing the frame transformation tree. This is built using the UR arm encoders and the platform tracking camera. While the former is highly accurate, the latter does fail in some cases. When platform position estimates are wrong, all targets in $\hat{y}$ get shifted and most will be treated as false positives and slowly removed. Because of the arm-in-hand configuration, all grasps are based on current camera frame observations, so this does not present a problem for actual harvesting. However, it does cause the farm scale fruit yield map to be inaccurate. Adding an absolute positioning sensor, such as global positioning system (GPS), would fix this in future.

Depth readings beyond a maximum or below a minimum will be inaccurate and are ignored. Camera motion is also checked when taking observations and detections during high linear or angular velocity are ignored. Inaccuracies from motion occur due to transformation tree lag, as this takes some time to update, and due to camera blurring. Each fruit object in $\hat{y}$ also has properties stored in a corresponding data structure, external to the EKF. This allows ripeness, health, size, visibility, and other traits relevant for growers, to be easily recorded for later use.

### 4.3.2 Extended Kalman Filtering

The extended Kalman filter framework is applied here to iteratively track fruit positions, but also later for camera pose estimation and in the context of active perception. To support these later sections, the general EKF form is first presented, then details specific to static fruit tracking are described. The state vector of fruit positions is denoted $x_k$, while $y_k$ is the camera pose state vector, both of these are modelled using a nonlinear function with additive noise

$$\begin{aligned} x_k &= f(x_{k-1}) + w_k \\ y_k &= f(y_{k-1}, u_k) + w_k^y \end{aligned} \tag{4.1}$$

where $u_k$ is a known input and the $k$ subscript is a discrete time index. This section uses $x_k$ to denote a generic system state vector, with $y_k$ later used for camera pose and $x_k$ for fruit pose. The transition function $f(\cdot)$ must be linearised when calculating filter gains, this Jacobian is denoted $A$ and defined for $x$ as

$$A_k = \left.\frac{\partial f(\cdot)}{\partial x}\right|_{x=\hat{x}_k} \tag{4.2}$$

with a similar definition for $y$ linearised at $y = \hat{y}_k$. The observation function $h(\cdot)$ is used to relate state vectors to sensor readings $z_k$ using

$$z_k = h(x_k) + v_k \tag{4.3}$$

where $w_k$ and $v_k$ are uncorrelated white Gaussian noise with corresponding covariance matrices $Q$ and $V$. The Jacobian of $h(\cdot)$ is denoted $H$. The filter initialisation $x_0$ is distributed as

$$x_0 \sim \mathcal{N}(\bar{x}_0, \Sigma_{1|0}) \tag{4.4}$$

With these models in place, the EKF prediction step, which occurs once per $k$ iteration, is used to forward predict the state vector and covariance matrix $\Sigma$ of this,

$$\hat{x}_k = f(\hat{x}_{k-1}, u_k) \tag{4.5}$$

$$\Sigma_k = A_{k-1}\Sigma_{k-1}A_{k-1}^T + Q \tag{4.6}$$

Filter innovations are the difference between the expected observations from the current state, and those received from sensors

$$\bar{\varphi}_k = z_k - \hat{z}_{k|k-1} \tag{4.7}$$

Likewise, state estimate error is calculated using

$$\varepsilon_k = x_k - \hat{x}_k \tag{4.8}$$

If an observation also occurs during a given prediction time step the EKF update step is run to calculate the filter gain $K$, this is used to update the state vector and covariance

$$K_k = \Sigma_k H_k^T (H_k \Sigma_k H_k^T + V)^{-1} \tag{4.9}$$

$$\hat{x}_k = \hat{x}_k + K\bar{\varphi}_k \tag{4.10}$$

$$\Sigma_k = (I - K_k H_k)\Sigma_k \tag{4.11}$$

This generic EKF form is applied to target tracking of 3D fruit locations by using $y_k$ as the filter state with known camera poses $x_k$. These are constructed as

$$\hat{y}_k = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & \dots & z_n \end{bmatrix}^T \tag{4.12}$$
$$x_k = \begin{bmatrix} X^C & Y^C & Z^C & \phi & \alpha & \psi \end{bmatrix}^T$$

for $n$ fruit being tracked, where $[x_1, y_1, z_1]$ are the world frame coordinates of the first fruit being tracked and $x_t$ defines the transform from the world to camera axes and is known from the robot arm joint encoders. With static targets Equation 4.1 becomes

$$\hat{y}_k = I_{3n,3n}\hat{y}_{k-1} \tag{4.13}$$
$$\Sigma_k = \Sigma_{k-1} + Q$$

Both $Q$ and $V$ are assumed static and the observation function $h(\cdot)$ is the pinhole camera model. In practice, $H$ is calculated using differentiation by complex parts. When a new detection $z_k = [x^C, y^C, z^C]^T$ is processed, target association occurs by comparing the pose to each element of $\hat{y}_k$. If the minimum Euclidean distance from $z_k$ to a point in $\hat{y}_k$ is less than a set threshold, it becomes associated with that state element. Updates occur using Equations 4.9 - 4.11, with $\hat{y}_k$ in place of $\hat{x}_k$.

As in Algorithm 4.1 each EKF update is done for a single target observation, $z_k$ is stacked $n$ time to form a vector matching $\hat{y}_k$ in size. The $K_k$ matrix has elements not corresponding to the current target index, found through the target association step, zeroed out. The following noise matrices are used, these change size with $\hat{y}_k$ as new targets are added.

$$Q = 0.01 \times I_{3n,3n} \tag{4.14}$$
$$V = 0.02 \times I_{2n,2n}$$
$$\Sigma_0 = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.1 \end{bmatrix},$$

The predict step of (4.5) and (4.6) is run once per RGBD frame, so is dependent on

the sensor frame rate. Variable loop rates are typically a problem for EKF filtering, but the static motion model in (4.13) means this is less of an issue. The update step occurs for each target observation, so false positive detections typically receive a single update, while easy to detect fruit are updated many times.

One disadvantage of this approach is the state vector length, and thus filter complexity, increases as new targets are observed. EKF steps are relatively efficient to begin with, so this was not a problem during field trials. Complexity in very large orchards can be managed using sub-maps, such as one per crop row.

Another limitation is the static target motion model with additive noise. Wind and trellis movement results in difficult to predict but consistent target motion with periodic components. Estimation of this could improve filtering performance in future but removes some of the beneficial properties of a static target motion model, including tolerance to EKF loop rate changes and those described in Section 4.5.

False positive detections are removed as a separate function once per RGBD frame, following each EKF update step. The predicted image frame position for each target $h(\hat{y}_t)$ is reused from (4.7). Most of these will fall outside the current camera frame, for the set of those which should be visible, but do not have an associated observation in that frame, an *unseen* counter is incremented. Targets with an *unseen* value above a limit are assumed to be false positives and removed from $\hat{y}$ and $\Sigma$. With a 10.5Hz frame rate, using the default limit value of 3, false positives are typically present for 0.3s.

## 4.4 Study: Improving Visual Servoing Using Autocovariance Least Squares

Selecting the noise matrices $\Sigma_0, Q, V$ when constructing an EKF is often done ad-hoc, with guessed values using trial and error. This leads to sub-optimal filter performance which makes pose estimates less accurate and can reduce grasping performance, as identified in Rajamani and Rawlings (2009). Autocovariance least squares is a recent

correlation based method for principled noise covariance (NC) estimation. Visual servoing has many robotics applications. In the harvesting prototype, IBVS is used for final gripper approach, while the target tracking EKF is a form of PBVS, which is also explored in Section 4.5 for active perception localisation. Better estimates for NC matrices could improve all of these system components. To investigate this, a study is conducted on applying the autocovariance least squares (ALS) method to PBVS. For simplicity, and to make the results as widely applicable as possible, the generic PBVS case for a known tabletop object is considered. While the harvesting system uses IBVS rather than PBVS for final approach control, and direct Euclidean pose filtering for localisation, the same ALS technique could be applied to both of these filtering tasks with minor adjustments to the state transition and observation models. This thesis section is a summary of work presented in Brown et al. (2019) and Brown et al. (2020), for more details please see these papers. Figures and tables in this section are also adapted from these.

In this study, ALS is applied to position based visual servoing pose estimation under the extended Kalman filter and moving horizon estimation frameworks, with Gaussian noise assumptions. The ALS method works by formulating a least squares optimisation problem which minimises the difference between the expected and actual innovation autocovariances. To do this, a sub-optimal filter with gain $L$ is constructed and run to generate $L$-innovations, along with steady state data. For an optimal Kalman filter the innovations sequence is white, see (Anderson and Moore, 2012, chp.9). However the $L$ filter is sub-optimal due to the NC choices, so the innovations will be self-correlated. Using the ergodicity of this process and the autocorrelations matrix, the least-squares optimisation problem is constructed according to Odelson et al. (2006). Selection of the initial $L$ gain will impact the NC estimate variance and Duník et al. (2017) provide a means of choosing $L$.

The PBVS estimation goal is to recover the camera pose given image frame detections of known points on an object, it can also be applied to the inverse case for object pose estimation. Filtering techniques are applied to compensate for noisy feature tracking data. Under the assumption of constant camera motion for each sample

interval, an LTI model can be used. The camera state in Equation 4.12 is augmented to a 12 dimensional vector of $x_t$, and the derivatives of that. This augmented state is denoted $\tilde{x}$. As in the previous section, the pinhole camera observation model is used. To apply an EKF, the object feature point projection is linearised about the current state estimate for each step. Interactive results files and code for this study are available.[9]

### 4.4.1 ALS Notation

Within the ALS study section $[a_1, \cdots, a_n]$ denotes the stacking of $a_n$ scalars, vectors or matrices as $[a_1^{\mathrm{T}} \cdots a_n^{\mathrm{T}}]^{\mathrm{T}}$. The property $(ABC)_s = (C^\top \otimes A)(B)_s$ for stacked matrices is used and $(\cdot)_{ss}$ is the column-wise stacked lower triangular elements of a symmetric matrix. As shown in Rajamani and Rawlings (2009), the full column rank duplication matrix $\mathscr{D}_r \in \mathbb{R}^{r^2 \times \frac{r(r+1)}{2}}$ can be constructed which contains only zeros and ones, and satisfies $(Q)_s = \mathscr{D}_r (Q)_{ss}$.

### 4.4.2 Preliminaries

The pinhole model from Section 3.9 is used for camera modelling. Frames of reference are defined as $(X^C, Y^C, Z^C)$, $(X^O, Y^O, Z^O)$, $(u^C, v^C)$ for the camera, object and image plane respectively.

For a point in the object frame $P^O = [P_X^O, P_Y^O, P_Z^O]^T$ the projection to a point $P^C = [P_u^C, P_v^C]^T$ in the camera $(u^C, v^C)$ frame is given by

$$\begin{bmatrix} P^C \\ 1 \end{bmatrix} = \boldsymbol{C} \begin{bmatrix} \mathbf{R}_{OC} & \mathbf{T}_{OC} \end{bmatrix} \begin{bmatrix} P^O \\ 1 \end{bmatrix} \tag{4.15}$$

where $\mathbf{T}_{OC}$ and $\mathbf{R}_{OC}$ are translation and rotation from the object to camera axes. A calibrated camera is assumed, so $\boldsymbol{C}$ is known, while object feature points $P^O$

[9] https://github.com/jaspereb/ALS_MHE_Results

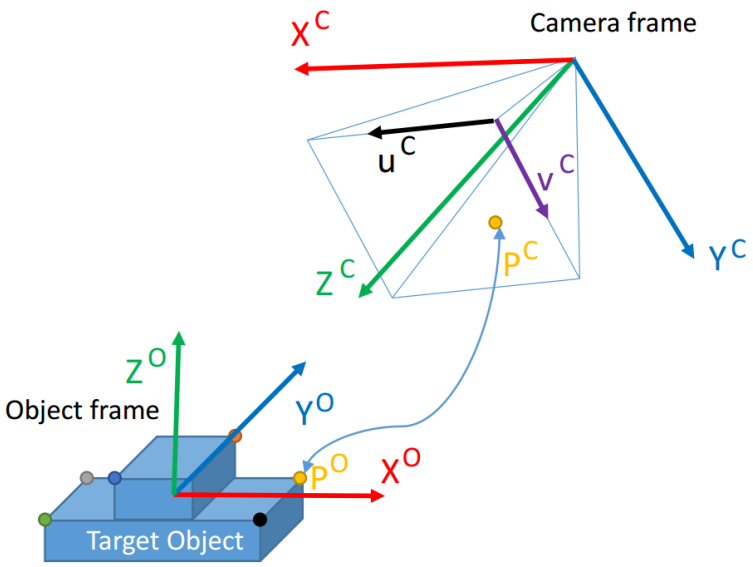


**Figure 4.9** – Projection of object feature points into the camera coordinate frame

come from a computer aided design (CAD) model and $P^C$ points are obtained as observations from the camera. With 6 unknown state parameters in $\mathbf{T}_{OC}$ and $\mathbf{R}_{OC}$, and 2 nonlinear equations to constrain these, a minimum of 4 non-collinier and non-coplanar feature points are required. Additional points will improve noise tolerance and increase the likelihood that more than 3 are visible from a given pose.

Unlike for the fruit tracking case, the target $y$ is assumed fixed at the world frame origin and the EKF estimates a 12 element camera pose $\tilde{x}$

$$\tilde{x} = \begin{bmatrix} X & \dot{X} & Y & \dot{Y} & Z & \dot{Z} & \phi & \dot{\phi} & \alpha & \dot{\alpha} & \psi & \dot{\psi} \end{bmatrix}^{\mathrm{T}} \tag{4.16}$$

$\hat{x}$ is used to signify the filter estimate of $\tilde{x}$. As in Wilson et al. (1996) and Janabi-Sharifi and Marey (2010), a step wise constant velocity model allows for a simple state transition model of

$$\tilde{x}_{k+1} = A\tilde{x}_k + w_k \tag{4.17}$$

for $k = 1, \cdots, M$ where $M$ is the total number of data points, and $A \in \mathbb{R}^{12\times 12}$ is the

block diagonal matrix

$$A = \begin{bmatrix} A_e & 0 & \dots & 0 \\ 0 & A_e & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \dots & A_e \end{bmatrix}, \quad A_e = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{4.18}$$

$T$ is the time interval between filter iterations. The non-linear measurement model comes from (4.15),

$$z_k = h(\tilde{x}_k) + v_k \tag{4.19}$$

with $z_k \in \mathbb{R}^{2p \times 12}$ where $p = 6$ is the number of feature points. The system initial state is assumed to be sampled as $\tilde{x}_0 \sim \mathcal{N}(\bar{\tilde{x}}_0, \Sigma_{1|0})$, with $w_k$ and $v_k$ being zero-mean uncorrelated Gaussian noise with corresponding covariance matrices of $Q$ and $V$. These are positive semi-definite and statistically independent from $\tilde{x}_0$. The set of $(\Sigma_0, Q, V)$ forms the EKF noise covariance matrices.

### 4.4.3 State Estimation for PBVS

The extended Kalman filter is a commonly used estimation method, being an efficient and easy to implement online technique. Kalman filtering is optimal for the linear case with Gaussian noise, but is formally the single Newton step version of full information estimation (FIE). Applying either FIE, or the windowed version of this, moving horizon estimation (MHE), will give better predictions of $\tilde{x}_t$. In this study, both estimation approaches are tested for generating the initial $L$-gain ALS data.

#### The Extended Kalman Filter

Building on the EKF definition from Section 4.3.2 the $L$-gains sub-optimal filter is constructed. With the actual initial state and NC matrices unknown, guessed

values for these $\hat{\tilde{x}}_{1|0}, \Sigma_g, Q_g, V_g$ are used, these are also inputs to the ALS process. $H$ is obtained by applying the linearisation method in (4.2) to $h(\cdot)$ to generate the approximate LTV model in discrete time

$$z_k \approx H_k(\tilde{x}_k - \hat{x}_k) + v_k \tag{4.20}$$

The initial $L$ gains filter can then be constructed using the EKF prediction and update steps in Equations 4.5 - 4.11. Following Rajamani and Rawlings (2009), the $\bar{A}_k, \bar{G}_k$ and $\bar{w}_k$ matrices are defined for convenience

$$\begin{aligned} \bar{A}_k &= A - AL_k^{so} H_k \\ \bar{G}_k &= \begin{bmatrix} I_n & -AL_k \end{bmatrix} \\ \bar{w}_k &= \begin{bmatrix} w_k^\top & v_k^\top \end{bmatrix}^\top \end{aligned} \tag{4.21}$$

Estimation error and observations can then be expressed in terms of realised noise values

$$\begin{aligned} \varepsilon_{k+1} &= \bar{A}_k \varepsilon_k + \bar{G}_k \bar{w}_k \\ \varphi_k &= H_k \varepsilon_k + v_k \end{aligned} \tag{4.22}$$

To apply the ALS method using this EKF, the $A$ and $H$ matrices are assumed uniformly detectable, and the filter is exponentially stable in $k = 1, \cdots, M$ meaning the expected value of the state estimate error goes to zero as k approaches infinity.

### Moving Horizon Estimation

While Kalman filtering is optimal for linear systems under Gaussian noise, more advanced techniques such as full information estimation perform better when these conditions do not hold. Computational complexity is a problem for FIE and increases with $M$, but by windowing the length of previous data considered, moving horizon

estimation can overcome this. Using the standard form in Ge and Kerrigan (2017) MHE can be described as

$$\Theta_{\bar{k}}^{*} = \min_{X_{k_s,\bar{k}}} \|\varepsilon_{k_s}\|^2_{\Sigma_{k_s}^{-1}} + \sum_{k=k_s}^{\bar{k}-1} \|w_k\|^2_{Q_g^{-1}} + \sum_{k=k_s+1}^{\bar{k}} \|v_k\|^2_{V_g^{-1}}, \tag{4.23}$$
$$\text{s.t.} \quad \tilde{x}_{k+1} = A\tilde{x}_k + G_k w_k,\ k = k_s, \ldots, \bar{k}, -1,$$
$$z_k = h(\tilde{x}_k) + H_k v_k,\ k = k_s, \ldots, \bar{k},$$

in which the data window length is $N_l$ and $k_s := \max\{\bar{k} - N_l, 0\} + 1$ with $\bar{k} = 2, ..., M$ being the current filter time step. The optimisation variable $X_{k_s,\bar{k}}$ is the sequence of states, which may have constraints imposed, within the data window. The $\Sigma_{k_s}$ values come from the previous EKF run in (4.11). The MHE estimator is used to generate the $X_{k_s,\bar{k}}$ state estimates which are applied to Equations 4.6, 4.7, 4.9 and 4.11 to generate the ALS input data $(K_k, H_k, \varphi_k)$.

### 4.4.4 The ALS Method for Noise Covariance Estimation in PBVS

For the above $L$ gains filter, the correlations of $\varphi_1, \varphi_2, \cdots, \varphi_M$ are non-zero and the auto-covariance of $\varphi_k$ over $j = 0, 1, \ldots, N-1$ time lags can be calculated as

$$\mathscr{C}_j(\varphi_k) = \mathbb{E}[(\varphi_{k+j} - \mu_{k+j})(\varphi_k - \mu_k)^\top] = \mathbb{E}[\varphi_{k+j}\varphi_k^\top] - \mu_{k+j}\mu_k^\top \tag{4.24}$$

which is presented in Odelson et al. (2006) and where $\mu_{k+j} = \mathbb{E}[\varphi_{k+j}]$. The ALS parameter $N$ is the maximum number of time lags considered.

For given observations $(z_k)_{k=1}^{\bar{k}}$ over the entire data window $\bar{k} \leq M$, assume the existence of an initial state $\hat{x}_{1|\bar{k}}$, where $\hat{x}_{1|\bar{k}} = \mathbb{E}\left[\tilde{x}_1 | (z_k)_{k=1}^{\bar{k}}\right]$ which can be found through smoothing. If this smoothed initial state is used for filter initialisation such that $\hat{x}_{1|0} = \hat{x}_{1|\bar{k}}$, then $\mathbb{E}[\varepsilon_1] = 0$.

The state error $\varepsilon_{k+1}$ is a function of $\varepsilon_1$ and $(\bar{w}_k)_{k=1}^{M}$. Because $\mathbb{E}[\varepsilon_1] = 0$, Equation 4.24 simplifies to

$$\forall\ k, j : \mu_{k+j} = 0 \Longrightarrow \mathscr{C}_j(\varphi_k) = \mathbb{E}[\varphi_{k+j}\varphi_k^\top] \tag{4.25}$$

For a window of innovations $(\varphi_{k+1})_{k=1}^{N_z}$, where $N_z = M_e - N + 1$, the lagged auto-covariance is

$$\mathscr{C}_j\left((\varphi_{k+1})_{k=1}^{N_z}\right) = \begin{bmatrix}\mathscr{C}_j(\varphi_2) & \cdots & \mathscr{C}_j(\varphi_{M_e-N+2})\end{bmatrix} \tag{4.26}$$

in which $M_e$ defines the data length used for estimation and it is assumed $N \ll M_e \leq M$. The auto-covariance matrix $\mathcal{R}$ can then be defined as

$$\begin{aligned}\mathcal{R} &= \begin{bmatrix}\mathscr{C}_0^\top\left((\varphi_{k+1}^\top)_{k=1}^{N_z}\right) & \cdots & \mathscr{C}_{N-1}^\top\left((\varphi_{k+1}^\top)_{k=1}^{N_z}\right)\end{bmatrix} \\ &= \begin{bmatrix}\mathcal{R}_0 & \mathcal{R}_1 & \cdots & \mathcal{R}_{M_e-N}\end{bmatrix}\end{aligned} \tag{4.27}$$

and

$$\mathcal{R}_i = \mathbb{E}\begin{bmatrix}\varphi_{2+i}^\top\varphi_{2+i} & \cdots & \varphi_{2+i}\varphi_{N+1+i}^\top\end{bmatrix}^\top \tag{4.28}$$

in which $i = 0, \cdots, M_e - N$. This auto-covariance matrix $\mathcal{R}$ is a function of $(\Sigma_1, Q, V)$ meaning the expected value of $\mathcal{R}$ for given $(\Sigma_1, Q, V,)$ can be determined. Using actual process measurements, the observed innovations $\bar{\varphi}$ can be calculated using (4.7) and the observed auto-covariance matrix $\bar{\mathcal{R}}$ is estimated from data as

$$\bar{\mathcal{R}} = \begin{bmatrix}\bar{\varphi}_2\bar{\varphi}_2^\top & \cdots & \bar{\varphi}_{M_e-N+2}\bar{\varphi}_{M_e-N+2}^\top \\ \bar{\varphi}_3\bar{\varphi}_2^\top & \cdots & \bar{\varphi}_{M_e-N+3}\bar{\varphi}_{M_e-N+2}^\top \\ \vdots & \ddots & \vdots \\ \bar{\varphi}_{N+1}\bar{\varphi}_2^\top & \cdots & \bar{\varphi}_{M_e+1}\bar{\varphi}_{M_e-N+2}^\top\end{bmatrix} \tag{4.29}$$

Generally, the $\varphi$ sequence coming from model (4.22) is non-stationary. However, with an ergodic process, $\bar{\mathcal{R}}$ can be accurately approximated from observations. By combining the analytic form of $\mathcal{R}$ from (4.27) with $\bar{\mathcal{R}}$ calculated from actual data, an unconstrained least squares problem can be established to estimate the true covariance matrices

$$(\widehat{\Sigma}^*_{1|0}, \widehat{Q}^*, \widehat{V}^*) = \arg \min_{\widehat{\Sigma}_{1|0}, \widehat{Q}, \widehat{V}} \left\| \mathcal{R}(\widehat{\Sigma}_{1|0}, \widehat{Q}, \widehat{V}) - \bar{\mathcal{R}} \right\|_F^2 \tag{4.30}$$

Ge and Kerrigan (2017) has shown that $\mathcal{R}$ can be composed as

$$\begin{aligned} \mathcal{R}(\Sigma_{1|0}, Q, V) = \Gamma\Big(I_{N_z} \otimes \Sigma_{1|0}\Big)\bar{\Gamma}^\top + \Omega\Big(I_{N_d} \otimes Q\Big)\bar{\Omega}^\top \\ + \Phi\Big(I_{N_d} \otimes V\Big)\bar{\Phi}^\top + \Psi\Big(I_{N_z} \otimes V\Big)\bar{\Psi}^\top \end{aligned} \tag{4.31}$$

The least squares problem in 4.30 requires vectorising $\bar{\mathcal{R}}$ and $\mathcal{R}$ as

$$\begin{aligned} (\mathcal{R}_i)_s = (\bar{\Gamma}_i \otimes \Gamma_i)\mathcal{I}_{1,n}(\Sigma_{1|0})_s + (\bar{\Omega}_i \otimes \Omega_i)\mathcal{I}_{i+1,r}(Q)_s \\ + [(\bar{\Phi}_i \otimes \Phi_i)\mathcal{I}_{i+1,q} + H_{i+2} \otimes \Psi_i](V)_s \end{aligned} \tag{4.32}$$

where the definitions for $\bar{\Gamma}, \Gamma, \bar{\Omega}, \Omega, \bar{\Phi}, \Phi, \bar{\Psi}$ and $\Psi$ are given in Rajamani and Rawlings (2007). $\bar{\mathcal{R}}$ is similarly vectorised as $\bar{b} = (\bar{\mathcal{R}})_s$. While (4.30) is an unconstrained optimisation problem, the positive semi-definiteness, a required property for accurate NC matrices, of $(\Sigma_1, Q, V)$ can be guaranteed by enforcing $\widehat{\Sigma}_{1|0}, \widehat{Q}, \widehat{V} \succeq 0$ during optimisation. Adding this constraint results in the central ALS problem

$$\min_{\vartheta} \left\| \mathscr{A}\vartheta - \bar{b} \right\|_2^2 \text{ s.t. } \widehat{\Sigma}_{1|0}, \widehat{Q}, \widehat{V} \succeq 0 \tag{4.33}$$

in which

$$\vartheta = \begin{bmatrix} (\widehat{\Sigma}_{1|0})_{ss} & (\widehat{Q})_{ss} & (\widehat{V})_{ss} \end{bmatrix}^{\mathrm{T}} \tag{4.34}$$

The definition of $\mathscr{A}$ is found in Ge and Kerrigan (2017). This can be solved as an SDP problem using CVX (Boyd and Vandenberghe, 2004; Grant and Boyd, 2013, chap. 4).

The ALS optimisation problem in (4.33) grows rapidly with $i$ due to the Kronecker products, quickly making it infeasible for large $M_e$. Memory efficient versions of this are developed in Ge and Kerrigan (2017). One approach uses the property $(A \otimes B)(C \otimes D) = AC \otimes BD$ from (Bernstein, 2009, chap. 7) to decompose the permutation matrices in 4.32 into smaller Kronecker product sums. Equation 4.32 becomes

$$\begin{aligned}(\mathcal{R}_i)_s = &\left[\sum_{j=1}^{i+1}\left(\bar{\Gamma}_i\zeta_j^{\bar{\Gamma}}\right)\otimes\left(\Gamma_i\zeta_j^{\Gamma}\right)\right]\mathscr{D}_n(\Sigma_{1|0})_{ss} \\ &+\left[\sum_{j=1}^{i+1}\left(\bar{\Omega}_i\zeta_j^{\bar{\Omega}}\right)\otimes\left(\Omega_i\zeta_j^{\Omega}\right)\right]\mathscr{D}_r(Q)_{ss} \\ &+\left[\sum_{j=1}^{i+1}\left(\bar{\Phi}_i\zeta_j^{\bar{\Phi}}\right)\otimes\left(\Phi_i\zeta_j^{\Phi}\right)+H_{i+2}\otimes\Psi_i\right]\mathscr{D}_q(V)_{ss}\end{aligned} \tag{4.35}$$

in which $(.)_{ss}$ is the stacking of symmetric matrix lower triangular entries and $\mathscr{D}$ is a duplication matrix with the property $(.)_s = \mathscr{D}(.)_{ss}$. Other symbols are defined in Brown et al. (2020).

### 4.4.5 Experiments and Results

#### Simulation Experiments and Results

A V-REP simulation is run of a Microsoft Kinect V1 mounted on a UR5 arm with a simple target object in frame, as shown in Figure 4.10. Five coloured markers are fixed at known locations and the arm follows a set trajectory. Ground truth and simulated camera frames come from V-REP, an HSV filter extracts centroid observations $z_k$ from the camera frames which then have noise added. Both a linear and non-linear trajectory are tested. The $L$ gains filter is constructed using a grid search over NC values to find the minimum mean state error.

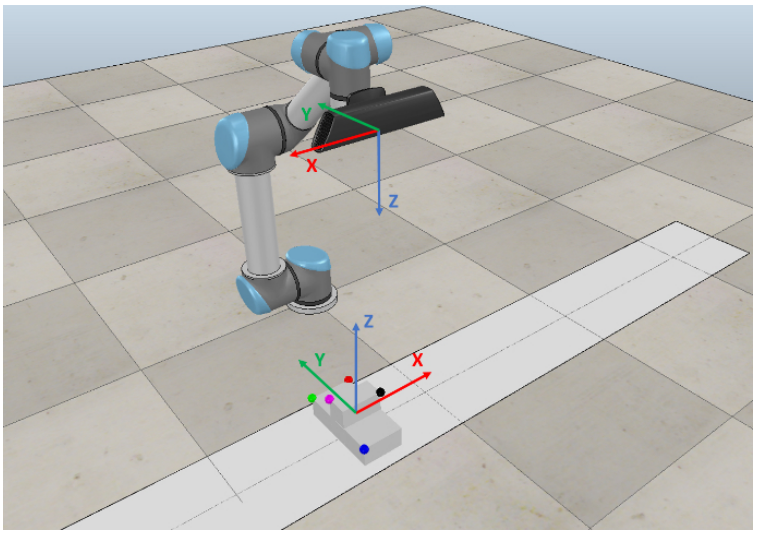


**Figure 4.10** – VRep simulation of the position based visual servoing scenario used to run simulated ALS experiments

Data lengths of $M_e = 70$ and $N = 20$ are used to construct $\bar{\mathcal{R}}$ in (4.29). To reduce the impact of $\Sigma_0$ choice, the first 50 time steps are discarded, allowing the filter to converge. Then the ALS optimisation in (4.33) is repeatedly solved for a moving window of length 50. All $M = 1450$ data points are covered by these windows, yielding 28 total segments, each with a separate $\widehat{Q}$ and $\widehat{V}$ estimate. The elementwise mean of these is calculated and used to construct the predicted optimal EKF, denoted the ALS-EKF. Exact values for the $L$ gain filter and ALS estimates of NC matrices can be found in Brown et al. (2020).

Diagonality constraints for $\widehat{Q}$ and $\widehat{V}$ are enforced by solving (4.33) as an semi-definite programming (SDP) problem. Additional structure can be easily enforced by specifying desired CVX constraints. Figure 4.11 displays the true states and both filter estimates for a segment of the first experiment. Numerical results for that experiment are in Table 4.3 which shows the ALS tuned EKF improved RMS position and angle errors by 33.1% and 50.5%, respectively.

The non-linear trajectory showed similar improvements to the first experiment as seen in Table 4.4, with position and angle estimate improvements of 38.5% and 34.5%. Violations of the constant velocity model appear as increased ALS estimates of the

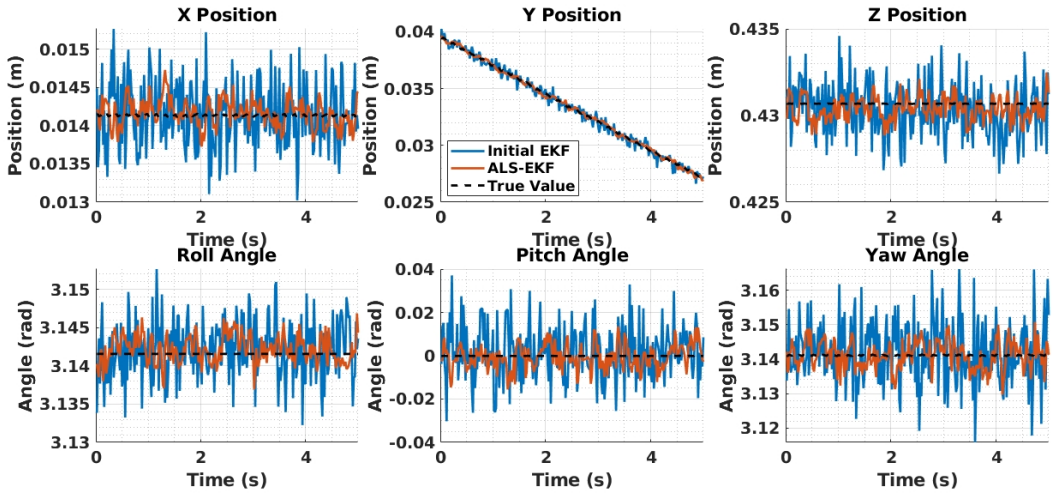


**Figure 4.11** – The estimated states and ground truth for the first 5 seconds of simulated linear trajectory test.

| Filter Type | $x$ (mm) | $y$ (mm) | $z$ (mm) | $\phi$ (mrad) | $\alpha$ (mrad) | $\psi$ (mrad) |
|---|---|---|---|---|---|---|
| | Mean Absolute Error | | | | | |
| $L$-gains EKF | 0.4026 | 0.3529 | 1.9380 | 3.6648 | 12.3844 | 8.9688 |
| ALS-EKF | 0.1919 | 0.1983 | 1.4119 | 1.9667 | 6.1530 | 4.2575 |
| | Max Error | | | | | |
| $L$-gains EKF | 1.3984 | 1.3797 | 6.5636 | 12.8461 | 43.0329 | 30.5378 |
| ALS-EKF | 0.7485 | 0.6775 | 3.4719 | 7.3749 | 27.3873 | 15.8724 |
| | Error Std Dev | | | | | |
| $L$-gains EKF | 0.4027 | 0.3513 | 1.5518 | 3.6476 | 12.3847 | 8.9711 |
| ALS-EKF | 0.1920 | 0.1935 | 0.7679 | 1.9531 | 6.1232 | 4.2535 |

**Table 4.3** – Estimation error for both filters on the linear simulation test.

system noise matrix $\widehat{Q}$.

An alternative measure of EKF efficacy is to examine the innovation sequence directly. For an optimal Kalman filter it is necessary and sufficient that this be white Gaussian noise. To determine the innovation whiteness, the pixel space innovations of the first marker are run through a Fourier transform, this occurs for both filters. As shown in Figure 4.13 the $L$-gains EKF exhibits a large low frequency peak, while the ALS-EKF data is closer to the ideal flat distribution.

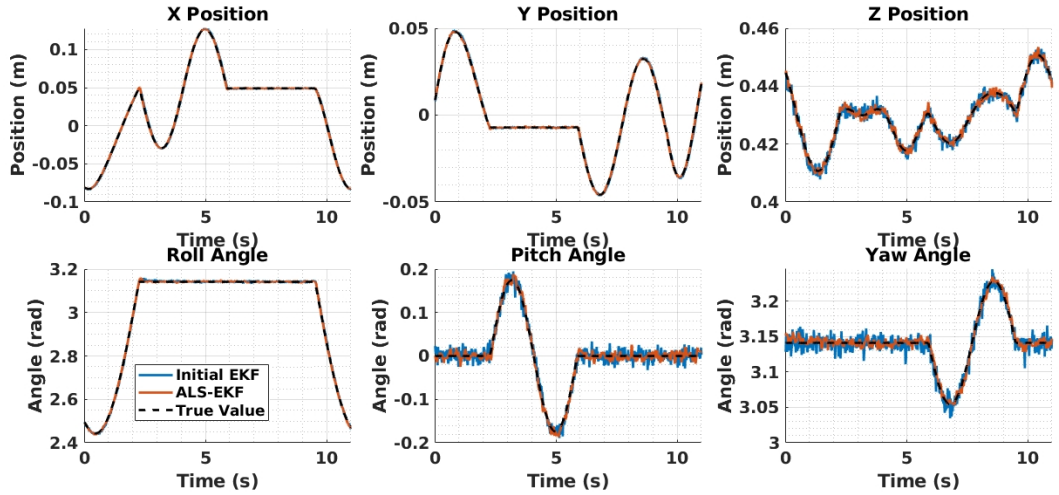


**Figure 4.12** – The estimated states and ground truth for the simulated non-linear trajectory test.

| Filter Type | $x$ (mm) | $y$ (mm) | $z$ (mm) | $\phi$ (mrad) | $\alpha$ (mrad) | $\psi$ (mrad) |
|---|---|---|---|---|---|---|
| | Mean Absolute Error | | | | | |
| Initial EKF | 1.3226 | 0.7634 | 1.9533 | 5.2877 | 11.9752 | 8.8000 |
| ALS-EKF | 0.5519 | 0.3701 | 1.5612 | 3.5728 | 8.2918 | 5.2000 |
| | Max Error | | | | | |
| Initial EKF | 4.6820 | 1.9997 | 6.3692 | 21.3880 | 41.8449 | 32.1000 |
| ALS-EKF | 4.1088 | 1.7287 | 4.9078 | 19.2321 | 32.8545 | 19.3000 |
| | Error Std Dev | | | | | |
| Initial EKF | 1.3230 | 0.7620 | 1.5826 | 5.2715 | 11.9418 | 8.8000 |
| ALS-EKF | 0.5520 | 0.3688 | 1.1690 | 3.5659 | 8.2676 | 5.3000 |

**Table 4.4** – Estimation error for both ALS study filters on the non-linear simulation test.

### Physical Experiments and Results

PBVS tuning using ALS is also assessed using a physical experiment. A checkerboard target of known size is imaged by a camera fixed to the UR5 arm. A 60 second trajectory consisting of many curved and linear motions with variable acceleration is run. To construct the ground truth, the 54 checkerboard corner points are localised in each image. Total reprojection error is used as the goal for a minimisation problem to determine camera pose for each frame, followed by a two-point moving average

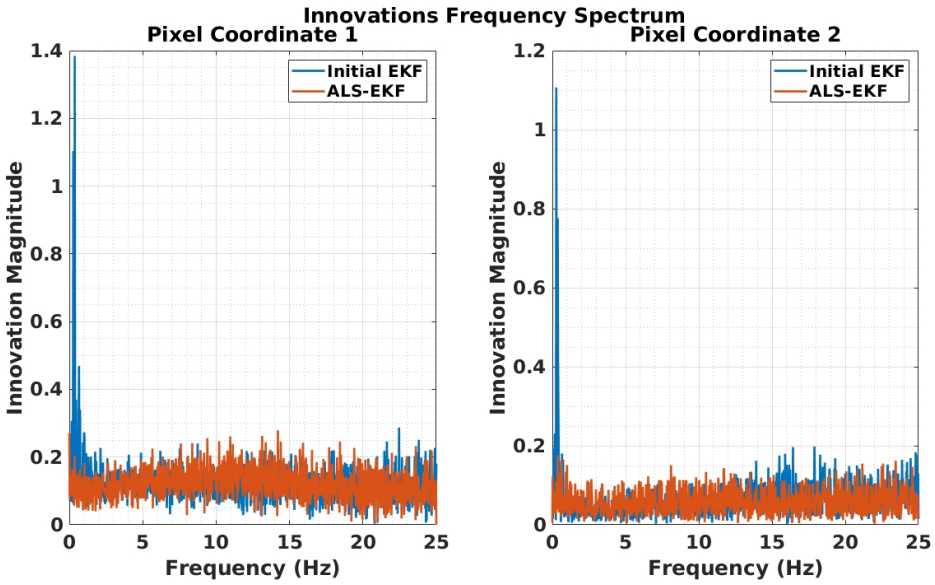


**Figure 4.13** – The frequency spectrum for the innovation sequence in the non-linear simulated trajectory experiment. Left and Right are the innovations of image pixel coordinates $u$ and $v$, respectively.

filter. The checkerboard centre point and 4 corners are used to build $z_k$.

For the given setup, ALS was found to be numerically sensitive to innovation magnitude. Very small time steps often result in $\widehat{Q}$ values collapsing to zero. The data frame rate is down sampled to prevent this by keeping one in $n$ sequential images. Conversely, over downsampling makes the ALS data window longer with greater violation of the constant velocity assumption and less accurate ALS estimates. A value of $n = 6$ is used, corresponding to 0.1 second time steps, practically this can be found by progressively downsampling data until $\widehat{Q}$ estimates of zero are no longer seen.

To select an $N$ value the innovation correlation coefficients are plotted in Figure 4.14. Selecting the first local minima around $N = 20$, appears to be a good experimental heuristic for $N$. When this extrema occurs is influenced by the time step length, actual system behaviour and $L$-gains filter NC guesses. A coarse grid search over $L$-gains EKF NC parameters resulted in the Table 4.5 parameters.

Similar to the simulation experiment, constant values are assumed for $\widehat{Q}$ and $\widehat{V}$, determined as the mean over sliding window estimates. ALS theory suggests that

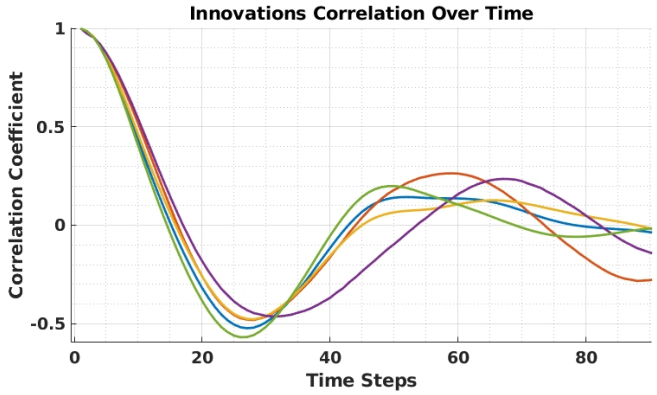


**Figure 4.14** – The correlation coefficient for innovations on the first 5 measurement vector elements, generated by the initial-EKF. The $x$ axis shows varying numbers of lagged time steps.

| Parameter | $N$ | $M_e$ | $n$ | $W$ | $N_l$ | $Q_g$ | $V_g$ | $\hat{x}_{1\|0}$ |
|---|---|---|---|---|---|---|---|---|
| Value | 20 | 21 | 6 | 28 | 4 | $10^{-6} \times I_{12}$ | $10^{-4} \times I_{10}$ | See Below |

**Table 4.5** – Key autocovariance least squares experiment parameters

$M_e \gg N$ will yield the best ALS estimate. However for this application the innovation correlations quickly decay to zero, as the constant velocity model is violated in inconsistent ways for longer time periods, meaning larger $M_e$ windows will only add noise and computational complexity. To balance the assumed constancy for $\widehat{Q}$ and $\widehat{V}$ with non-informative correlations over longer windows, a small window length of $M_e = N + 1 = 21$ was used and found to be experimentally effective.

Using a small value for $M_e$ also has computational advantages but causes larger variance in the per-window ALS estimates, sometimes resulting in unstable EKF filters constructed from these. An additional constraint is introduced to prevent this by enforcing

$$\begin{aligned} \widehat{Q}_{1,1} &= \widehat{Q}_{3,3} = \widehat{Q}_{5,5} \\ \widehat{Q}_{2,2} &= \widehat{Q}_{4,4} = \widehat{Q}_{6,6} \\ \widehat{Q}_{7,7} &= \widehat{Q}_{9,9} = \widehat{Q}_{11,11} \\ \widehat{Q}_{8,8} &= \widehat{Q}_{10,10} = \widehat{Q}_{12,12} \end{aligned} \tag{4.36}$$

The initial state estimate used for all tests is

$$\hat{x}_{1|0} = \begin{bmatrix} 0.08 & 0 & 0.03 & 0 & 0.32 & 0 & 0 & 0 & 0 & 0 & -\pi & 0 \end{bmatrix}^\top \tag{4.37}$$

Generating the data used for ALS estimation is a separate problem from applying ALS NC estimates for filtering, the first of these steps can be done using an EKF or MHE. Online filtering requires a very efficient framework, and is not always possible with MHE, so only Kalman filtering is tested for second step. The filter constructed using $L$-gains EKF data is denoted EKF-ALS-EKF. While the one which uses MHE to generate the ALS data, then estimates the NC matrices using ALS and applies these to an EKF is denoted MHE-ALS-EKF.

Initial state conditions are important to MHE performance, so a two step process is used. First, MHE is run using the value from (4.37). Then $\tilde{x}_{1|0}$ estimated from that MHE run is used to initialise a new MHE estimate of the data. This new MHE run generates the ALS innovations. All moving horizon estimate optimisation problems use a horizon length of $N_l = 4$, which is less than the ALS parameter $N$. Experimental assessment determined that ALS estimates for $\widehat{V}$ produced worse results than using the $V_g$ value, so $V_g$ is used in place of the ALS $V$ estimate for the results presented here. This discrepancy may be caused by how the ground truth is generated, which uses independent per-frame estimates from the same noisy camera sensor as the filter data, but with additional points for increased accuracy.

State estimates generated by the $L$-gains EKF, EKF-ALS-EKF and MHE-ALS-EKF are shown in Figure 4.15 and enumerated in Table 4.6. Using MHE estimates as input to the ALS optimisation improved final filter performance by a further 21% over the

EKF-ALS-EKF case, for a total improvement of 48% over the grid search filter.

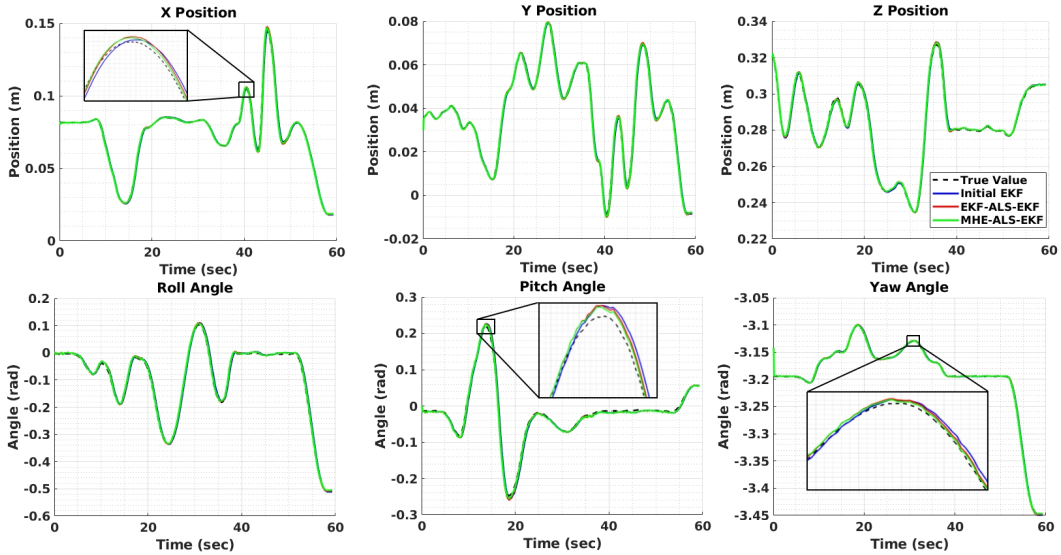


**Figure 4.15** – The real experiment data estimated states for the initial filter and both ALS tuned filters, plus ground truth.

Benefits of the efficient ALS implementation are reflected in Table 4.7. The memory use and computation time were significantly reduced, but ALS remains an offline method for this scale of problem. MHE is also time intensive for this data length. Reported total time includes the generation of results figures and running the ALS tuned EKF.

### 4.4.6 ALS Study Conclusion

Results obtained from simulation and physical testing show that applying ALS for noise covariance matrix estimation can improve filter performance on the tested PBVS task. Using more accurate initial filters, such as moving horizon estimation, to generate ALS input data, further increased the accuracy of NC estimates. Computational complexity was effectively minimised using the sliding window approach with an efficient ALS implementation, while the constant velocity assumption keeps model complexity low. A heuristic for selection of $M_e$ and $N$ is also proposed.

| Filter Type | $x$ (mm) | $y$ (mm) | $z$ (mm) | $\phi$ (mrad) | $\alpha$ (mrad) | $\psi$ (mrad) |
|---|---|---|---|---|---|---|
| | Mean Absolute Error | | | | | |
| $L$-gains EKF | 1.200 | 1.080 | 1.103 | 5.895 | 5.560 | 2.523 |
| EKF-ALS-EKF | 0.861 | 0.689 | 0.688 | 4.445 | 4.689 | 0.806 |
| MHE-ALS-EKF | 0.597 | 0.475 | 0.586 | 3.956 | 4.116 | 0.587 |
| | Max Error | | | | | |
| $L$-gains EKF | 7.225 | 3.581 | 4.010 | 17.345 | 15.676 | 4.767 |
| EKF-ALS-EKF | 4.172 | 1.785 | 2.052 | 10.279 | 12.780 | 2.648 |
| MHE-ALS-EKF | 2.963 | 1.430 | 1.766 | 10.339 | 8.982 | 2.040 |
| | Error Std Dev | | | | | |
| $L$-gains EKF | 1.199 | 1.081 | 1.075 | 4.857 | 4.992 | 2.496 |
| EKF-ALS-EKF | 0.862 | 0.689 | 0.623 | 3.057 | 3.991 | 0.777 |
| MHE-ALS-EKF | 0.597 | 0.474 | 0.511 | 2.412 | 3.291 | 0.562 |

**Table 4.6** – Estimation error for all three ALS study filters on the real experiment data.

| **For the efficient ALS method with $N = 20$** | | | |
|---|---|---|---|
| Configuration | $M_e = 21$ | $M_e = 70$ | $M_e = 110$ |
| Original ALS Time (sec) | 0.69 | 45 | 131 |
| Efficient ALS Time (sec) | 0.68 | 24 | 41 |
| Original ALS Memory (GB) | <0.01 | 4.73 | 13.6 |
| Efficient ALS Memory (GB) | <0.01 | 0.4 | 0.6 |
| **For the entire estimation process with 28 windows** | | | |
| Configuration | Filtering Time (sec) | ALS Time (sec) | Total Time (sec) |
| EKF-ALS-EKF | 0.9 | 21.9 | 26.3 |
| MHE-ALS-EKF | 824.0 | 14.3 | 839.9 |

**Table 4.7** – Autocovariance least squares computation requirements

The NC matrices are assumed to be static, a reasonable but inaccurate simplification given a constant velocity model. Many camera trajectories, including those in industry, can be partitioned into repetitive segments. During harvesting, the motion can be delineated into approach, retraction and drop steps. Different ALS NC estimates could be developed in future for each of these motions by aligning the moving data window with each motion segment. Even with reduced complexity, ALS remains an offline method. Some numerical stability issues related to innovation magnitude were

seen during optimisation.

While the ALS method could provide better NC estimates for the fruit localisation EKF, performance of that filter using guessed $Q, V, \Sigma$ matrices was already sufficient. During the field trial, no pick failures could be attributed to EKF performance, so development effort was instead dedicated to less reliable system modules, such as the object detection framework. As the fruit tracking process is improved in the future through better detections, filter performance may become limiting and ALS will be considered to improve this. However, determining ground truth to quantify improvement for the fruit tracking filter is difficult and time consuming.

## 4.5 Study: Active Perception

The methods described above all require depth information to localise the 3D position of each fruit. This is provided by the D435i stereo camera, which is not always robust to outdoor lighting conditions. Estimation of target poses can instead by done using bearings-only observations from multiple image frames, as an arm mounted 2D camera approaches the fruit. If used, this technique would allow continuous 3D tracking when the depth map is inaccurate or unavailable and would improve harvesting system robustness. This is a basic form of structure from motion, where the image features to match are object detections and the target geometry is known, so only pose needs to be estimated. Many techniques exist to solve this task, including particle filtering and full information estimation. However, the EKF framework is both iterative, allowing for easy online updates of target positions as the camera moves, and has an explicit analytical form for the filter uncertainty, namely, the state covariance matrix.

Under the pinhole camera model, each image frame constrains the target to lie along a ray joining the camera pose to the detection centroid in the optical frame. For an accurate camera and spherical target of unknown size, the unique intersection of two rays is sufficient and necessary to locate the target in 3D space, as shown in Figure 4.16. In practice there will be estimation noise due to errors in the camera pose,

object detection centroid, pixel discretisation and target movement, so additional observations are required to reduce estimate error.

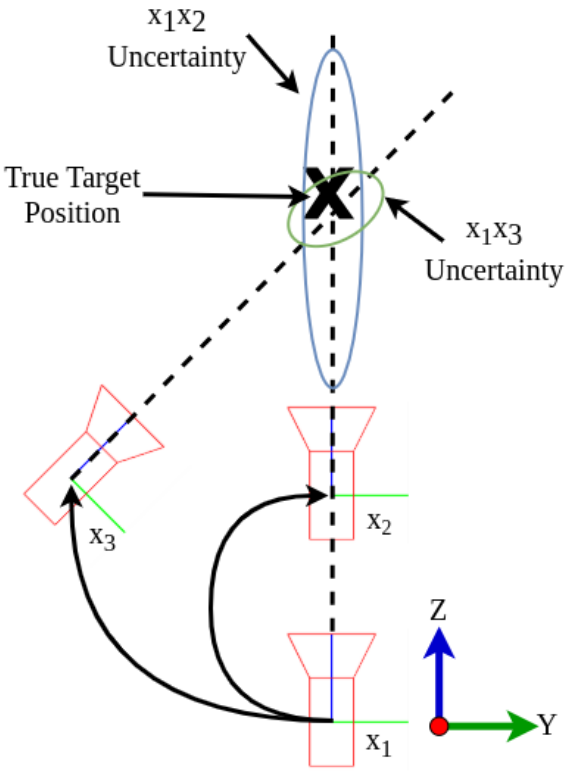


**Figure 4.16** – Illustration of the AP problem, looking at the top view of a scene with two possible camera paths $x_1x_2$ and $x_1x_3$. The first of these reduces uncertainty in the Y and X dimensions but not Z (blue ellipse), while the latter path trades some Y uncertainty for much lower Z uncertainty (green ellipse). In this scenario there is a prior estimate of the target Z range, otherwise the blue ellipse would be infinitely long.

For an in-hand camera which approaches a fruit along the optical $z$ axis, typically the most direct path to the target, coincident rays are produced which cannot be used to estimate the fruit depth $z$. Instead, a camera path should be selected which provides a large angular separation between rays to properly estimate $z$, while keeping the target in frame. This goal must be balanced with the desire to execute a minimum time trajectory which ends with the gripper over a target fruit. Choosing an optimal sensor trajectory for fruit localisation and harvesting is an active perception (AP) problem, and techniques from this field can provide a principled approach. Code for

this study is available online.[10]

### 4.5.1 Problem Formulation

The problem of choosing an optimal sequence of camera-in-hand gripper poses to maximise grasp success is considered in both the offline, and online case. Following the framework, and substantially the notation, in Atanasov et al. (2014) and Schlotfeldt et al. (2018), models of the camera motion, target motion and predicted observations are defined. The sensor state model is given by

$$\begin{aligned} x_{t+1} &= f(x_t, u_t) \\ u_t &\in \mathcal{U} \end{aligned} \tag{4.38}$$

where $x_t = [x, y, z, qw, qx, qy, qz]$ is the camera pose at time $t$, consisting of the 3 DoF position and an orientation quaternion. $u_t$ is a single action step drawn from the set of actions $\mathcal{U}$. This is a constant, 5 element set of position offsets which move the camera towards the initial target pose estimate in the pattern shown in Figure 4.17.

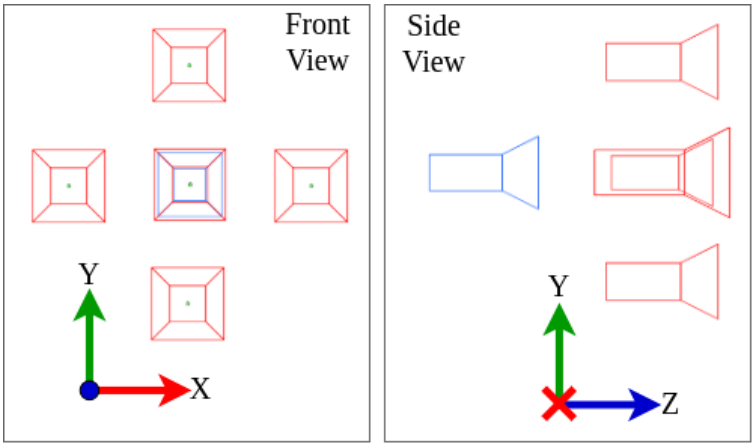


**Figure 4.17** – The set of 5 possible next poses (in red) from a given pose (in blue). The z axis points to the target position estimate $\hat{y}$.

[10] `https://github.com/jaspereb/AP_Experiments`

At each camera position, the orientation quaternion is calculated to point it towards the target estimate $\hat{y}$, meaning the state space complexity is reduced from 6 degrees to 3. By using a fixed target location and generating these on a regular grid, calculating the node dominance criteria for reduced value iteration (RVI) is simplified. After applying these simplifications and removing dominated nodes, the number of nodes at a given timestep grows quadratically, rather than exponentially. This is important for tractability in longer sensor paths.

All experiments are performed for a single target, although the framework is easily extended to multi-fruit tracking once a target association function is in place. The fruit targets are assumed to have an identity motion model, making them static with no additive noise

$$y_{t+1} = I_3 y_t \tag{4.39}$$

$\hat{y}_t$ is the current estimate of the fruit centroid location. The pinhole model, described in Section 3.4.1, is used to generate sensor observations according to

$$z_t = h(x_t, y) + v_t \tag{4.40}$$

$$v_t \sim \mathcal{N}(0, V) \tag{4.41}$$

where $z_t$ is the fruit pixel coordinates. The sensor noise is modelled as constant magnitude pixel noise, independent of the sensor or target states. Object detector inaccuracy was the primary noise source observed for fruit position estimation, and is influenced by camera blur and target size, which is not captured in the noise model. Incorporating these effects would increase the model accuracy and is an avenue of future work. Including the transformation from world to camera coordinates, the pinhole model is

$$h(x_t, y) = \mathcal{C} f_{prj}(\boldsymbol{R}_t^T(y - p_t)) + v_t \tag{4.42}$$

$$f_{prj}(y) = \frac{1}{y_3}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \tag{4.43}$$

where $p$ is the first 3 elements of the camera state $x$, while $\boldsymbol{R}$ is the camera orientation expressed as an $SO(3)$ rotation matrix. The projection function $f_{prj}$ is required to linearise the observation function in the target coordinates. $\mathcal{C}$ is the following modified form of the camera intrinsic matrix

$$\mathcal{C} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \end{bmatrix} \tag{4.44}$$

Forward value iteration FVI of the state covariance matrix requires the observation model to be linear in the target state, so $h(.)$ is linearised about the current target estimate. For the offline case

$$z_t \approx H(x_t)y + v_t \tag{4.45}$$

$$H(x_t) = \nabla_{y=\hat{y}_0} h(x_t, y) \tag{4.46}$$

and for the online case

$$H(x_t) = \nabla_{y=\hat{y}_t} h(x_t, y) \tag{4.47}$$

this linearisation for the pinhole camera model is given in Schlotfeldt et al. (2019) as

$$H(x_t) = \mathcal{C} f'_{prj}(\boldsymbol{R}_t^T(y - p_t))\boldsymbol{R}_t^T \tag{4.48}$$

$$f'_{prj}(y) = \frac{1}{y_3^2}\begin{bmatrix} y_3 & 0 & -y_1 \\ 0 & y_3 & -y_2 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.49}$$

### 4.5.2 Optimisation Goal

Choosing a set of actions to maximise the final mutual information between the target state and observations is the overall active perception goal. For fruit harvesting, an additional termination constraint $x_T = \hat{y}_{T-1}$ is introduced to place the final sensor pose, and thus the gripper position, at the target. When operating online, the information available to plan sensor actions includes previous actions and observations

$$\begin{aligned} \mathcal{I}_0 &= z_0 \\ \mathcal{I}_t &= (z_{0:t}, \mu_{0:(t-1)}) \end{aligned} \tag{4.50}$$

given a static target model, the AP goal is

$$\begin{aligned} &\max_{\mu_0,\ldots,\mu_{T-1}} \mathbb{I}(y; z_{1:T}|x_{1:T}) \\ s.t. \quad &x_{t+1} = f(x_t, \mu_t(\mathcal{I}_t)), \quad t = 0, \ldots, T-1 \\ &x_T = \hat{y}_{T-1} \\ &z_t = H(x_t)y + v_t, \quad t = 0, \ldots, T \end{aligned} \tag{4.51}$$

Under the approximation of a linearised observation model and Gaussian noise assumptions, Atanasov et al. (2014) shows that this maximisation goal reduces to a deterministic optimal control problem which can be optimally solved offline. The following goal is equivalent to Equation 4.51 for any monotone concave cost function, such as the covariance matrix trace

$$\min_{\sigma \in \mathcal{U}_T} \text{trace}(\Sigma_T) \tag{4.52}$$
$$s.t. \quad x_{t+1} = f(x_t, \sigma_t), \quad t = 0, \ldots, T-1$$
$$\Sigma_{t+1} = \rho^p(\rho^e_{x_{t+1}}(\Sigma_t)), \quad t = 0, \ldots, T$$

where $\sigma$ is a camera trajectory, $\rho^e_x$ is the Kalman filter covariance update step

$$\rho^e_x(\Sigma) = (\Sigma^{-1} + M(x))^{-1} \tag{4.53}$$
$$M(x) = H(x)^T V^{-1} H(x)$$

and $\rho^p$ is the Kalman filter covariance prediction step, which is identity in this case

$$\rho^p(\Sigma) = A\Sigma A^T + Q = I\Sigma \tag{4.54}$$

Applying this simplification to (4.52) yields the $\Sigma$ update

$$\Sigma_{t+1} = (\Sigma_t^{-1} + H(x)^T V^{-1} H(x))^{-1}, \quad t = 0, \ldots, T \tag{4.55}$$

Both trace$(\cdot)$ and log det$(\cdot)$ have been suggested as generic cost functions to reduce final target uncertainty. However, the goal of this active perception step is to increase grasp success and not all $\Sigma$ components contribute equally to this. For this reason, a weighted trace cost function is proposed here to match target uncertainty to the gripper positioning tolerance, this results in the optimisation problem

$$\min_{\sigma \in \mathcal{U}_T} \text{trace}(W_\Sigma \Sigma_T) \tag{4.56}$$

where $W_\Sigma$ is the target pose estimate covariance weighting matrix

$$W_\Sigma = \begin{bmatrix} W_{\Sigma_x} & 0 & 0 \\ 0 & W_{\Sigma_y} & 0 \\ 0 & 0 & W_{\Sigma_z} \end{bmatrix} \tag{4.57}$$
$$s.t. \quad W_{\Sigma_x}, W_{\Sigma_y}, W_{\Sigma_z} \geq 0$$
$$W_{\Sigma_x} + W_{\Sigma_y} + W_{\Sigma_z} = 1$$

This is assumed to be analytically or empirically estimated from gripper geometry so that sampling the target location according to this distribution maximises grasp success probability

$$\max_{W_\Sigma} p(\text{grasp success}|x_T, y) \tag{4.58}$$
$$s.t. \quad x_T = \hat{y}$$
$$\hat{y} \sim N(y, W_\Sigma)$$

While $W_\Sigma$ will have off-diagonal elements in practice, these are difficult to estimate directly from gripper geometry. Learning these from large numbers of harvest trials is a potential area of investigation.

### 4.5.3 Reduced Value Iteration & Kalman Filtering

Planning an optimal sensor trajectory for harvesting a specific fruit requires solving (4.52) and a forward value iteration (FVI) tree is the most straightforward means of doing this. At each camera pose the set of possible actions is iterated over. Each action yields a new node with camera pose determined by (4.38) and $\Sigma$ by (4.55). However, the FVI tree has a large amount of redundant nodes, where multiple nodes have equivalent $x_t$ but different $\Sigma_t$. By removing dominated nodes a still optimal RVI tree is constructed. Algorithm 4.2 describes this process.

The RVI algorithm builds a full tree where each node contains $x, \Sigma$ and also a pointer to its parent node. This allows $\sigma$ to be reconstructed from a given end node $N_T$.

**Algorithm 4.2:** Reduced Value Iteration Tree

**Input:** $x_0, \hat{y}_0, \Sigma_0$
**Output:** $N_{0:T}$
$N_0 \leftarrow (x_0, \Sigma_0)$;
$\mathcal{U} = \text{generateActionSet}(x_0, \hat{y}_0)$;
**for** $t = 1 : T$ **do**
  **forall** $(x, \Sigma) \in N_{t-1}$ **do**
    **forall** $\mu \in \mathcal{U}$ **do**
      $x_t \leftarrow f(x, \mu)$;
      $\Sigma_t = \rho^e_{x_t}(\Sigma)$;
      **forall** $(x', \Sigma') \in N_t$ **do**
        **if** $x' = x_t$ **then**
          **if** *costFunction(* $\Sigma_t$*)* $<$ *costFunction(* $\Sigma'$*)* **then**
            `/* The node is dominating */`
            $(x', \Sigma') \leftarrow (x_t, \Sigma_t)$;
            **break**
          **else**
            `/* The node is dominated */`
            **break**
        `/* The state is not yet visited */`
        $N_t \leftarrow N_t \cup (x_t, \Sigma_t)$

Because no actions move away from the target position $\hat{y}_0$, node state overlap will only occur for successive tree levels $N_t$ and $N_{t\pm1}$. This simplifies node dominance checking. With the removal of dominated nodes, only a single tree node will exist where $x_T = \hat{y}_0$. So an optimal, but not necessarily unique, grasping path can be recovered by finding this terminal node. Because the entire tree is enumerated in Algorithm 4.2, the lowest cost node ignoring the $x_T = \hat{y}_0$ constraint can also be found.

Maximum and minimum sensing distances are a key factor in real world depth camera performance. These are incorporated into the RVI tree by only running $\rho^e_x(.)$ to update the covariance when the $z$ distance in the camera FoR, from $p$ to $\hat{y}$ is between 0.25m and 4m. This same rule is applied when simulating observations for the EKF, using the distance $p$ to $y$.

Time horizon $T$ is set using a fixed number of steps, $T = 11$ for all experiments. The *generateActionSet* function determines the vector $\vec{u} = \hat{y}_0 - x_0$ and calculates regular steps along this

$$\begin{aligned} \vec{u} &= \hat{y}_0 - x_0 \\ \delta_z &= \frac{\vec{u}}{T} \\ \delta_y &= \delta_z \times \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \\ \delta_x &= \delta_z \times \delta_y \end{aligned} \tag{4.59}$$

where $\times$ denotes the vector cross product. By axis definitions, the world z axis points from $x_0$ to the target position $y_0$, before noise is added to that. The action offsets are then normalised and used to construct the 5 elements of $\mathcal{U}$ previously shown in Figure 4.17

$$\begin{aligned} \delta_y &= \frac{\delta_x |\vec{u}|}{|\delta_y| T} \\ \delta_x &= \frac{\delta_x |\vec{u}|}{|\delta_x| T} \\ \mathcal{U}_1 &= [0, 0, \delta_z] \\ \mathcal{U}_2 &= [0, \delta_y, \delta_z] \\ \mathcal{U}_3 &= [0, -\delta_y, \delta_z] \\ \mathcal{U}_4 &= [\delta_x, 0, \delta_z] \\ \mathcal{U}_5 &= [-\delta_x, 0, \delta_z] \end{aligned} \tag{4.60}$$

The EKF update step $\rho^e_{x_t}(\cdot)$ is applied to the predicted $\Sigma$ when generating the RVI tree. For this experiment, the process of actually moving a camera along the RVI determined trajectory is then simulated, and an EKF applied to mimic the real fruit localisation task. The same basic EKF approach as Section 4.3.2 is applied with the initial values and noise matrices of

$$x_0 = [0, 0, 0, 1, 0, 0, 0]^T \tag{4.61}$$

$$\Sigma_0 = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.25 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0.001 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0.001 \end{bmatrix}$$

$$V = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

While $\hat{y}_0$ is generated by projecting the initial detection $z_0$ to a mean trellis distance of 4m. For the offline case, an RVI tree is constructed once for each run using $x_0, \hat{y}_0, \Sigma_0$ then applied with the EKF. Performance of both the optimal constrained end-point $x_T = \hat{y}_{T-1}$ and unconstrained path are reported for the offline case. Building a full RVI tree from each new observation forwards is too inefficient for online arm motion planning, so the *online* RVI approach is a compromise intended to determine the impact of linearisation error.

To do this, the online method re-linearises the information transition model at each node of the RVI tree by taking an EKF measurement there. Thus each RVI node contains not only the $\Sigma$ and initial state estimate, but also a history of EKF observations to reach that node and updated state estimate from these. The online method is only practical in simulation because the camera must visit every RVI node, but is a fast approximation of how well true online RVI would perform.

### 4.5.4 Estimating $W_\Sigma$

Gripper geometry is analysed to generate the cost weighting matrix $W_\Sigma$. For simple grippers, such as the parallel gripper described Section 5.1.1 this can be done by inspection. The soft gripper from that section would need to be tested experimentally

to determine its tolerance to mis-positioning. The parallel gripper finger geometry, shown in Figure 4.18, will be used to estimate $W_\Sigma$ for all experiments and grips occur with the axis orientation shown in that figure. The $W_\Sigma$ matrix is a weighting factor and thus dimensionless, but is normalised so trace$(W_\Sigma) = 1$.

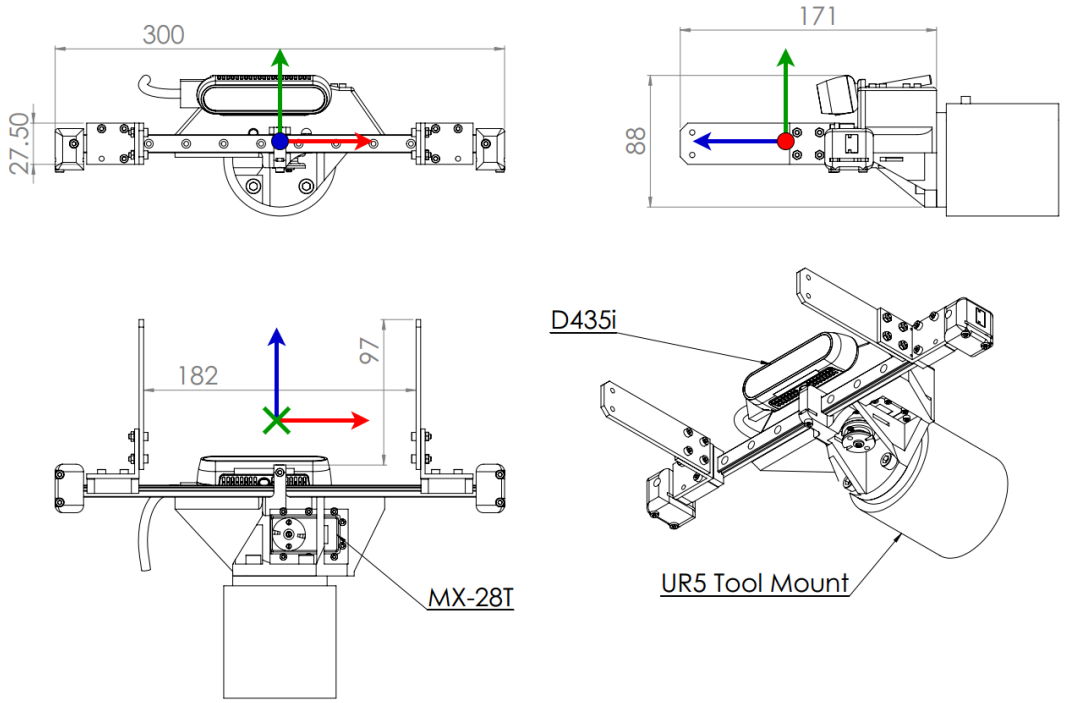


**Figure 4.18** – Dimensions of the fingers and base for the parallel gripper in millimetres, also axis definitions for all picks located at the ideal grip point, the convention of red-green-blue corresponding to X-Y-Z axes is used. Further design details in Section 5.1.1

In the vertical dimension, grips where the fruit centroid falls outside the top or bottom edges of the finger are likely to fail as the spherical fruit has insufficient friction to be held on the convex surface. Horizontally, the fingers are 182mm apart when fully open and most fruit positions within this will be successful. In the depth direction, fruit are stopped by the gripper base but will be damaged if pushed too far by this. Based on qualitative observations from preliminary field trials, the maximum push distance before damage occurs is roughly 30mm for this gripper when applied to apples. This is expected to be close to equivalent for similar sized crops, such as plums, but requires confirmation through further testing. For this study the depth

range, which is the push distance plus the finger length, is set to 127mm and the grip point used for planning should be 63.5mm in from the finger tips. For this gripper design, the y axis positioning is much more important to grasp success than either x or z. This analysis yields a $W_\Sigma$ in mm of

$$W_\Sigma = \begin{bmatrix} 182.0 & 0 & 0 \\ 0 & 27.5 & 0 \\ 0 & 0 & 127 \end{bmatrix} \tag{4.62}$$

which is elementwise inverted and normalised to

$$W_\Sigma = \begin{bmatrix} 0.110 & 0 & 0 \\ 0 & 0.731 & 0 \\ 0 & 0 & 0.158 \end{bmatrix} \tag{4.63}$$

Several scenarios were observed during later testing where off diagonal $W_\Sigma$ elements would be required to describe picking behaviour. One illustrative example is a fruit against one of the open fingers. As the gripper closes, pushing it along the $x$ axis, that fruit behaves as a pendulum and is also pulled upwards along an arc. This induces correlation between the $W_{\Sigma_x}$ and $W_{\Sigma_y}$ elements. Determining the magnitude of these values is difficult.

### 4.5.5 Active Perception Experiments

To simulate AP informed picking and compare it to manually programmed paths, six total path types are tested. These consist of two hand engineered trajectories, plus the offline and online RVI paths both with and without end point constraints. The offline constrained path corresponds to harvesting, while online and unconstrained path types are included to explore the impact of these limitations. Each of the 6 paths is run 1000 times with target locations generated around $y = [0, 0, 4]$ with additive noise of standard deviation 0.5m for the x, y dimensions, and 0.1m for z.

These six paths are assessed by running the EKF step on each and analysing the actual evolution of $\Sigma$ and $y - \hat{y}_t$.

The first hand engineered trajectory tested is a straight line, with equally spaced camera poses between $x_0$ and $\hat{y}_0$. This provides some low quality information about fruit depth as the additive target noise means $\hat{y}_0 \neq y$ and the fruit is not directly aligned with the straight path. The diagonal hand engineered path is v shaped and is chosen to better gather depth information while still being a relatively direct trajectory.

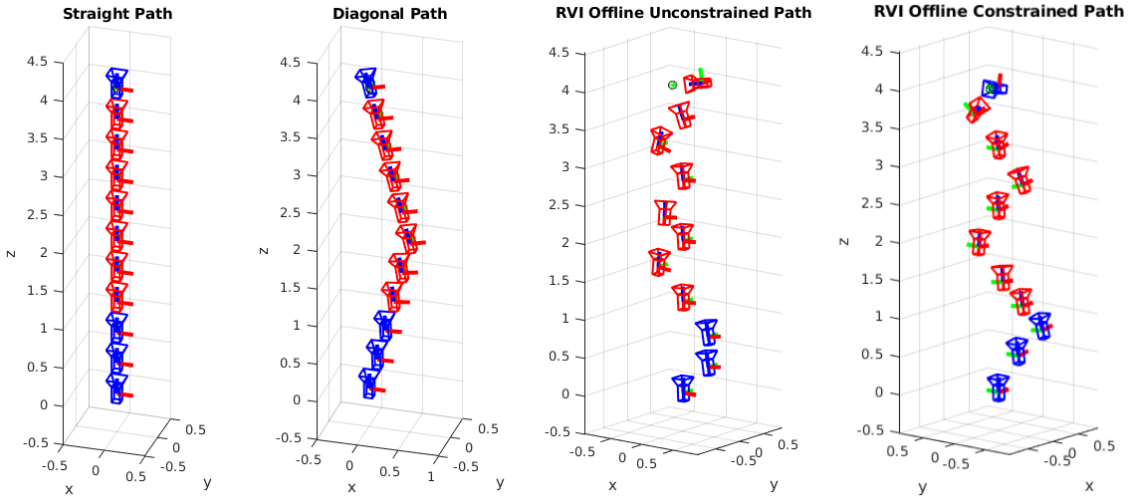


**Figure 4.19** – The straight line, and diagonal hand engineered sensor trajectories. Plus samples of constrained and unconstrained paths generated using online RVI. Blue indicates poses where observations cannot be taken, the green star shows the true target location and black circle the $\hat{y}_0$ estimate.

### 4.5.6 Active Perception Results

Over both hand chosen paths, plus offline and online RVI, the EKF state estimate errors and covariances are plotted in Figure 4.20. Figure 4.21 shows the weighted cost function for each path and Table 4.8 summarises this data. All values are calculated using 1000 EKF runs and are the covariance and cost realised by the filter, rather than predicted during RVI.

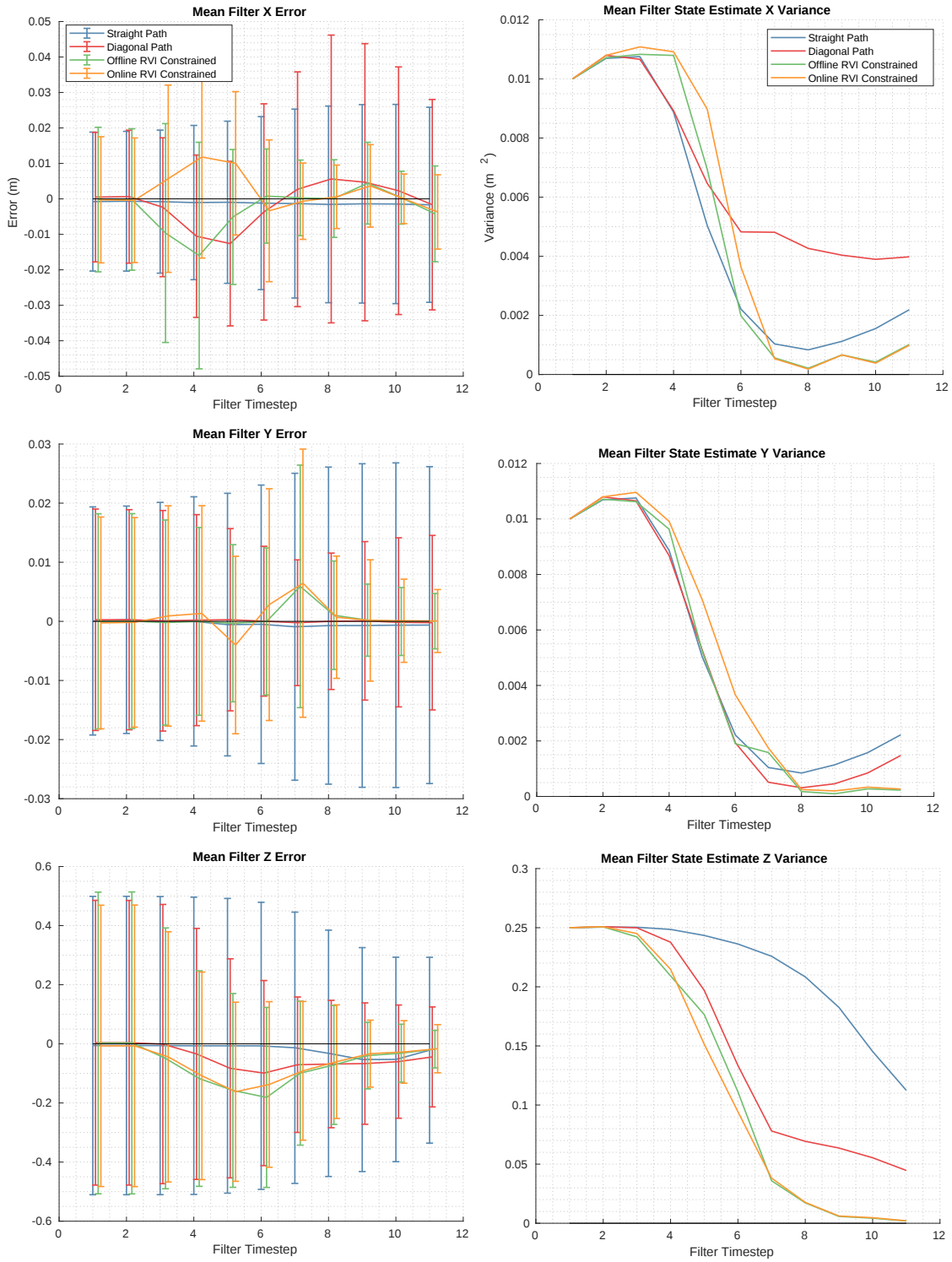


**Figure 4.20** – The mean EKF state estimate error with standard deviation (left), and mean EKF state estimate variance (right) for each path generation method. Paths are slightly offset on the x axis for presentation clarity.

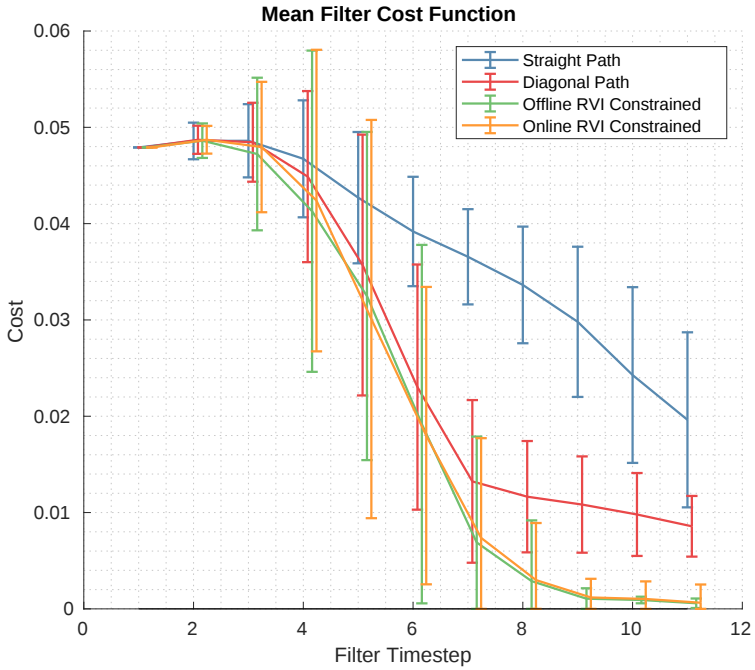


**Figure 4.21** – The weighted cost function value by pose number for each path generation method.

| Path Type | Mean Final Estimate Error (m) x | y | z | Mean Final Estimate Variance ($m^2$) x | y | z | Cost |
|---|---|---|---|---|---|---|---|
| Straight Path | -0.001663 | -0.00063 | -0.021539 | 0.027503 | 0.026807 | 0.314392 | 0.019628 |
| Diagonal Path | -0.001634 | -0.00022 | -0.044232 | 0.029666 | 0.014758 | 0.16909 | 0.008574 |
| Offline RVI Constrained | -0.004211 | 0.000039 | -0.017913 | 0.013504 | **0.004686** | **0.063592** | 0.000582 |
| Offline RVI Unconstrained | **0.000490** | -0.000105 | -0.023812 | **0.006374** | 0.005267 | 0.076789 | 0.000922 |
| Online RVI Constrained | -0.003673 | 0.000062 | -0.016535 | 0.01048 | 0.005322 | 0.081423 | 0.000635 |
| Online RVI Unconstrained | -0.003607 | **-0.000007** | **-0.015045** | 0.010458 | 0.004752 | 0.064933 | **0.000579** |

**Table 4.8** – Active perception experiments results, with minimum values highlighted.

From Table 4.8 it is observed that applying active perception significantly improved performance over both hand engineered paths. While mean EKF error is reduced by only 4% between the straight path and unconstrained online RVI, x dimension accuracy is traded off to decrease error in the highly weighted y axis by 99%. Mean covariance reduced by 75%. This is reflected in the final weighted cost where unconstrained online RVI also performs best.

Constraining the offline RVI path resulted in a lower cost function than unconstrained

offline RVI. Initial target pose estimate error is the likely cause for this. Because the constraint is applied to $\hat{y_0}$ which has significant added noise, the probability of having uninformative final poses is similar for the constrained and unconstrained case, making this distinction meaningless. The online method constrains the path using the final target pose estimate which has been updated over time, so does not face this issue. Final cost is slightly lowered by not applying end point constraints in the online version of RVI, as expected.

Examination of Figure 4.20 shows how both RVI cases take non-greedy actions which trade off x,y uncertainty to reduce z uncertainty. This results in a lower final variance in all 3 dimensions. Figure 4.21 also reflects this where cost values for intermediate path segments are highly variable and frequently worse than the diagonal path, but both result in low cost with low variance at the final time step. The increase at the start and end of these graphs comes from poses where the target is outside the sensing range and no EKF observations are made.

Both RVI approaches are run single threaded in Matlab with no code optimisation for speed. Calculated over 1000 runs, the EKF takes 27ms per run, with online and offline RVI requiring 8.84s and 6.50s respectively. With some improvements the RVI operations would be sufficiently fast for real-time picking motions.

### 4.5.7 Active Perception Study Conclusion

Simulated experiments clearly demonstrate that active perception techniques can be applied to generate informative paths for bearings-only fruit localisation using an imaging sensor and object detection with minimum and maximum sensing ranges. Applying constrained online reduced value iteration improved the y estimate standard deviation from 2.68cm for the straight path to 0.48cm for RVI. This change in error variance is the difference between the parallel gripper often failing, and accurately gripping a fruit almost all of the time.

Applying RVI in place of hand engineered trajectories is highly beneficial, however using the depth sensing data from the stereo camera will still produce better esti-

mates where this is available. Additionally, these simulated results fail to incorporate many of the issues seen with the RGBD camera sensing modality, including obscurations, detector failures, and target association challenges. Taking more complex paths through the trellis canopy will also result in additional collisions, which is the primary reason active perception is not applied to the prototype system. Faster collision detection and recovery would alleviate this, and AP techniques for picking guidance, with collision triggered online replanning in place, should be further explored.

Future directions of this study would also involve conducting physical experiments where position estimate accuracy can be related to empirical grasp success rates. Real experiments would also allow $W_\Sigma$ to be estimated from data, without constraints on diagonality. The Manhattan-distance type action space is overly simplistic and should be replaced by one with constant Euclidean distance between RVI nodes, but this negates the easy dominance criteria of the current action space resulting in significantly slower computation. Active perception may also be applied to determine fruit ripeness or health by selecting the best sensing modality.

# Chapter 5

# Grasping In Plum Crops

Upon successfully localising the target fruit, a grasp for this must be planned and executed. For the task of harvesting, the fruit also needs to be detached from the tree and dropped into a bin or processing system. Gripper design, actuator selection and picking motion all contribute to the success of this process, so various options are considered. One property that sets harvesting apart from many other robotic problems is the presence of mixed hard and soft obstacles. This complication is examined and addressed using a mix of soft robotics and careful planning constraints implemented using a picking state machine. Figures in this chapter are adapted from Brown and Sukkarieh (2021).

## 5.1 Gripper Design For Harvesting

Gripper geometry and robustness have significant impacts on harvesting success and several designs were tested prior to settling on two alternatives for the field trials. Many forms of end effector have been proposed for fruit harvesting, as reviewed in Davidson et al. (2020). Though a wide range of designs are proposed, one identified taxonomy of grippers is between pneumatic and electric actuation. Soft robotics is well matched to pneumatic actuation and, with this in mind, a rigid electric servo driven gripper is compared to a pneumatic soft gripper in field trials. Distributing

stress concentration over each spherical fruit is required to avoid bruising, this suggests the use of 3 or more fingers, or conformable components such as soft pads. Initially, two tendon driven designs with 3 fingers were developed, but these designs were very complex resulting in them being overly time consuming to fabricate and maintain, so a simple parallel gripper was eventually developed for further field testing. None of these three rigid designs address obstacle collisions, so a soft pneumatically actuated gripper was built and used for the final field trials. All four designs used 3D printing, rapid prototyping techniques and commercial parts where possible to minimise cost and development time.

Prior to harvesting system development, a copy of the Yale OpenHand Model O from Ma and Dollar (2017), was built for testing on generic agricultural manipulation tasks. Experience with this highlighted the benefits of under-actuation for organic geometries and the simplicity of tendon driven joints compared to geared or belt drive transmissions. The benefits of being able to re-configure the gripper size for various fruit types and the need for embedded sensing were identified.

Such functionality motivated the design of a modular 3 finger gripper with additional degrees of freedom which is shown in Figure 5.1. This was designed to support functionality beyond plum harvesting, such as tree stem spraying and handling small farm tools. These application goals resulted in a design with 18 degrees of freedom, including 6 for the base pose. It used embedded range sensing with similar ToF sensors to Section 3.4, while keeping the tendon drive approach of the OpenHand. Modular finger units were designed so that fingers could be easily added or removed, while the extra DoF on each module base allowed the finger separation to be dynamically reconfigured for large or small objects. This gripper was overly complex for harvesting, leading to planning complexity, robustness and construction time issues. While the hardware was fabricated and validated, it became apparent this design was not appropriate for harvesting before full DoF planning and embedded sensing feedback could be tested. Designs for autonomous harvesting must be very simple and robust to meet cost constraints and the high cycle rates.

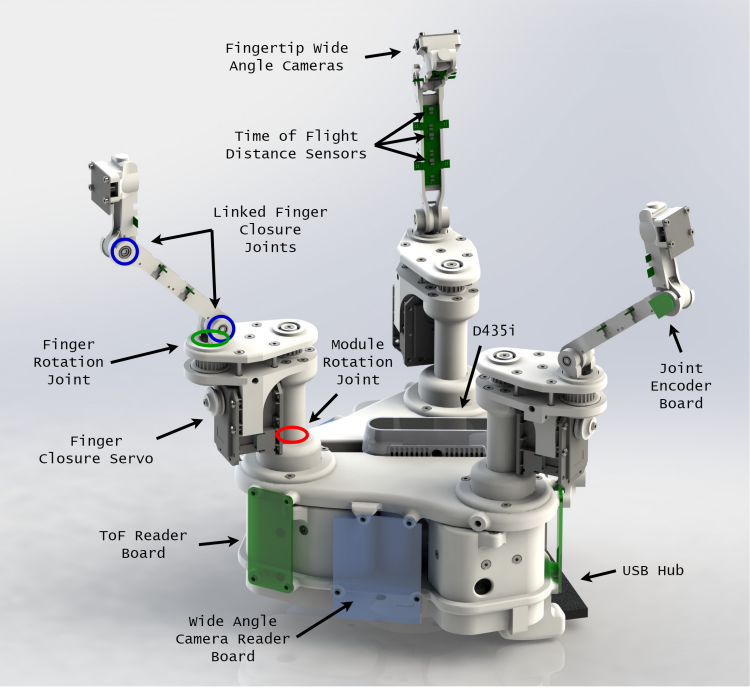


**Figure 5.1** – The high DoF re-configurable gripper design, shown with 3 independently actuated finger modules in the standard equilateral triangular configuration. Each colour denotes a single actuator controlling those joints.

### 5.1.1 Parallel Gripper

To improve robustness, the simplicity of a tendon drive design was applied to a low component count and easy to fabricate parallel gripper, shown in Figure 5.2. This was designed from scratch for plum harvesting with the specifications in Table 5.1. Two finger cars slide on a standard linear rail, with 3D printed components at the centre and ends of the rail. This means the gripper can be resized for different fruit by using longer or shorter rails, without changing any other parts. A single MX-28T servomotor turns a pulley which the continuous tendon loop is wrapped around, so that operation in one direction opens the fingers, and the opposite closes them. The pulley can be resized to alter the motor gearing for heavier and more robust fruit, or small and easily damaged targets. Force control and position feedback are also implemented on the MX-28T. Designs and software for this are made available

online.[11]

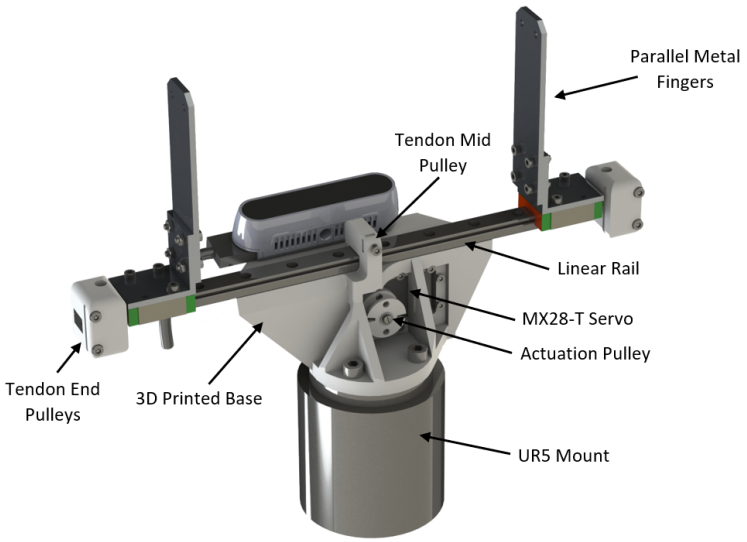


**Figure 5.2** – The simple parallel gripper used for field testing. The tendon (not shown) runs in a loop from one car, through the end pulley, to the actuation pulley via the mid pulley, then on to the other end pulley and car.

Finger plates for the parallel gripper bolt onto the sliding cars, so can easily be modified for various harvesting conditions. A finger size of 100 x 27.5mm, fabricated from 3mm aluminium was used for testing. This has a small amount of flex which helps regulate grasping force on the fruit, under full load, a finger tip deflection of approximately 15mm was observed. Cup type fingers were tested, but led to a large number of collisions when accessing fruit. A design trade-off exists for harvesting where sharp and narrow fingers are best for deflecting off hard obstacles, while broad flexible fingers are required to avoid bruising fruit. Achieving form closure of the fruit is only possible by placing finger segments on the far side of each target, the same part of the fruit which is least accessible.

Both the parallel and soft grippers are connected using their respective control boxes, this allows for an identical robot operating system (ROS) action service and hardware interface between the two. Control occurs using an Arduino with position and torque feedback used to determine if an object has been grasped. The parallel gripper closes

[11] https://github.com/jaspereb/SimpleSliderHand

| Property | Value |
|---|---|
| Tested Rail Length | 300mm |
| Max Rail Length | 600mm |
| Min Rail Length | 140mm |
| Max Finger Distance | Rail Length – 120mm |
| Min Finger Distance | 8mm |
| Max Force | 200N |
| Force Control Resolution | 0.2N |
| Max Actuation Time | 2.1s |
| Positioning Resolution | 0.02mm |
| Mass | 850g |
| Dimensions | 300x170x65mm |
| Tested Finger Dimensions | 100x27.5x3mm |

**Table 5.1** – Parallel gripper specifications, speed and force can be balanced by altering the tendon pulley diameter, while the rail length can be adjusted for different fruit

until a software configurable motor torque is reached.

### 5.1.2 Soft Gripper

Operating with both soft and hard obstacles is a major challenge, as described in Section 5.2. Several concepts were explored to design a gripper capable of recovering from collisions and deflecting over obstacles caught under the fingers when retrieving fruit. This is required to avoid fingers frequently becoming stuck on branches and stems near the target fruit. Placing springs in series with tensioning tendons would allow finger deflection, but physically incorporating sufficient length springs to avoid plastic deformation is challenging. Static friction and spring pre-loading also lead to slower and less predictable actuation. Flexible urethane joints, as in the OpenHand, are a hybrid of hard and soft fingers, but are not fully backdrivable due to inextensible tendons. Fast feedback force-torque sensing for collision detection is a control based solution, but is high complexity and requires an equally fast actuation response, which is not guaranteed with the current arm interface method.

Entirely soft fingers provide excellent tolerance to collisions, with rapid and simple actuation using compressed air. The downsides of pneumatic actuators, such as needing to determine their shape at design time and their inability to provide positioning feedback, are lesser drawbacks for the harvesting task than for other robotic

manipulation applications. Soft and organically shaped fruit are a good match for soft robotics components, with little chance of bruising. Small finger size and a fully separated power source allow multiple fingers to be positioned in a compact gripper with space for embedded sensing.

The fingers were designed and manufactured by Yi Sun as described in Sun et al. (2013) and Sun et al. (2017). Four of these are integrated into a basic cup-shaped gripper, illustrated in Figure 5.3, with the D435i mounted above them, and wide angle camera embedded in the gripper centre. All fingers are connected to a shared air loop at either ambient pressure or 60psi when actuated, so that they close together. The number and placement of fingers is experimentally optimised for plum-sized fruit. Too few fingers lead to the fruit not being enveloped and it escaping through the gaps between them. Mounting hardware complexity and probability of failures in the common air loop increase with the number of fingers, they also begin to collide with the sides of each other as radial spacing is reduced. Actuation control is binary with no feedback mechanism.

Three of the numerous stable gripper modes can be seen in Figure 5.3. The partially enclosed modes occur when an object is too large or sufficiently far away, stable mode (b) is reached by rotating the gripper during closing, while (c) occurs when the target is slightly below the grip point.

Mounting the D435i depth sensor above the fingers prevents it from contacting most fruit during harvesting, with a semi-rigid flap providing further protection. The embedded wide angle camera is positioned to provide a good view of the grasping area, while being sufficiently deep that fruit can be properly contained in the gripper cup without touching it.

## 5.2 Mixed Obstacle Planning

Soft, or moveable, obstacles are those a gripper can move through without resulting in damage to the system, these include leaves, vines, and small branches. Contact

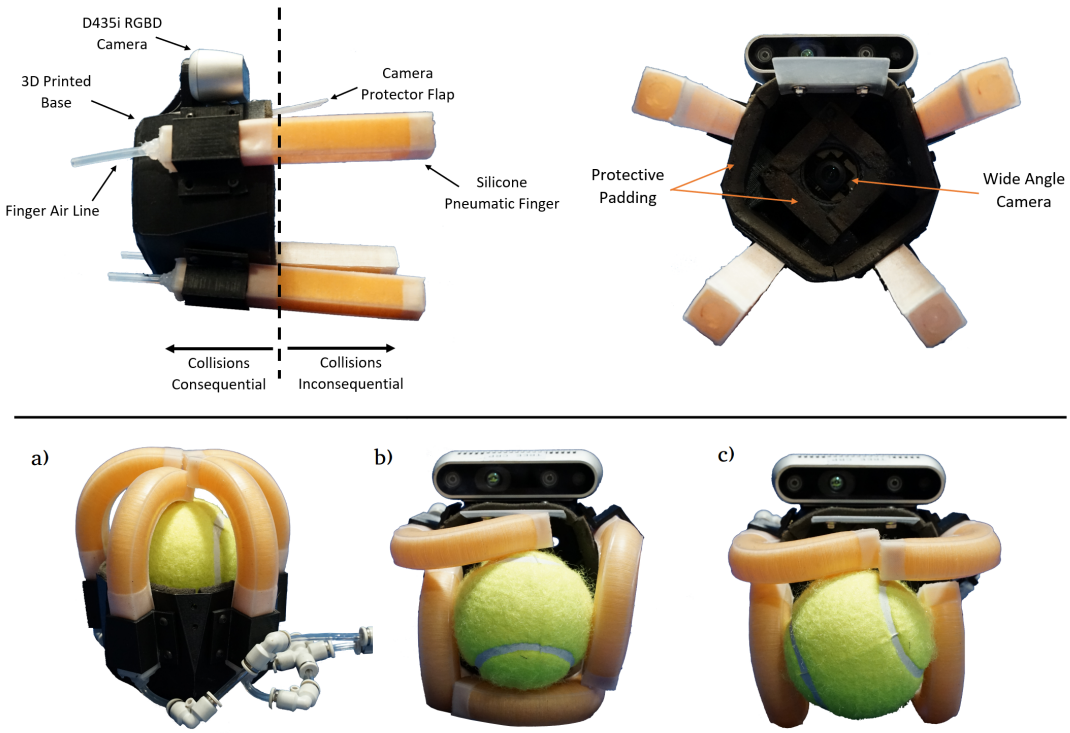


**Figure 5.3** – The four fingered soft robotics pneumatic powered gripper, with embedded sensing. Also shown are the hard and soft regions of the gripper, where collisions are consequential and inconsequential respectively. Three common stable finger configurations are shown; a) fully enclosed object, b) partially enclosed with rotation applied, c) partially enclosed with vertical movement applied.

with these may cause minor damage to the plant or crop but is often necessary for accessing fruit. Hard obstacles, such as trellis wires or posts, branches, and the ground, sometimes result in damage to the hardware. Discerning a hard from soft obstacle is challenging, hard trellis wires are much smaller than soft branches, while hard trunks are often obscured by soft leaves.

Building a perception system to classify obstacles into hard and soft varieties was considered, but adds significant complexity. Non-penetrating sensors, such as cameras, are inherently incapable of directly identifying obscured hard obstacles, including branches hidden by leaves. Both lidar and radar can penetrate leaves and may be applied for this in future, though considerable work is required to integrate and use these sensors for obscured obstacle reconstruction. Instead, the problem of mixed

**Figure 5.4** – Example frame with hard obstacles highlighted in red and soft obstacles in blue. Part of the gripper and one fingertip can be seen in the foreground.

obstacle planning can be solved at a system level. The harvesting prototype system does this by combining the inherent structure of the trellis system, with soft robotics components and constrained motion. Doing so avoids the need for additional sensors and perception algorithms.

The fruiting wall trellis structure happens to be excellent for automation, but was originally adopted for mechanisation. Because tractors, spray units, flower thinners, and other wide machinery must travel down each crop row several times per season, any hard obstacles beyond a certain distance from the trellis are either pruned off or broken by passing tractors. This effectively guarantees a plane, beyond which no hard obstacles are encountered.

When using the soft gripper described in Section 5.1.2 the plane beyond which no hard obstacles occur can be offset by the soft finger length, because contact between soft components and hard objects is acceptable. This results in the three planes shown in Figure 5.5. The approach motion region uses the approach and retraction arm controllers described below. Motion outside this is planned using RRTConnect and KDLKinematics for inverse kinematics, running in the MoveIt! framework with

unconstrained arm motions. Multithreaded processing is used for planning future motions while executing the current path. Repeated trajectories such as moving from the drop point to home pose are stored for reuse and executed at higher speeds.

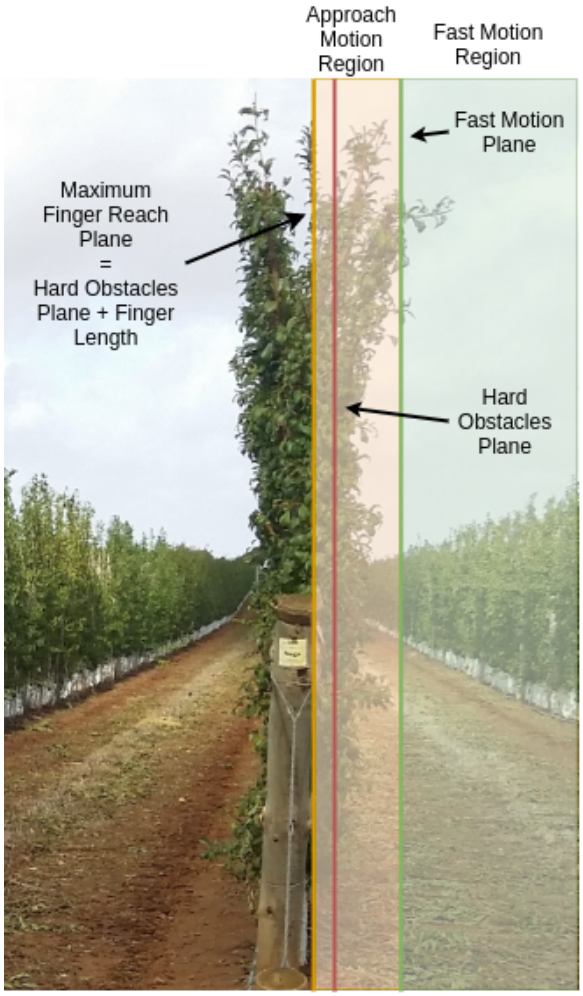


**Figure 5.5** – The planning planes and motion regions for harvesting the right hand side of a crop row. From right to left is the fast motion plane where no hard obstacles occur and full motion arm planning is used. Followed by the hard obstacle plane which rigid gripper components will not cross and where approach controller motions are used. Finally, the maximum finger reach plane, which is the deepest fruit reachable by the soft gripper components.

By leveraging the inherent trellis structure, soft robotics components and multiple constrained planning planes, damaging collisions are avoided without the need for highly complex perception steps. This approach to mixed obstacles does introduce

limits on other system design aspects. Specifically, only simple approach and retraction controllers can be used, preventing advanced motions such as choosing to pick fruit from above or below. It also requires a gripper positioning system capable of moving linearly from the outermost plane to the grip point, meaning no singularities can be present. The UR5 arm can only do this for small regions of motion, hence a restricted harvesting region of interest (RoI) measuring 0.5 x 0.5 x 0.8m is defined and only fruit within this are attempted. Figure 3.4 shows the RoI in blue. As mentioned below, the UR5 is only used for prototyping motions and a cartesian gantry would not suffer from the same RoI restriction.

## 5.3 Actuator Selection

Despite these issues with possible singularities in the approach and retraction controller motion, the choice of an articulated robot arm for gripper positioning was beneficial for testing other complex motion strategies. Use of a cartesian gantry system was also considered for its better speed, cost, weight and planning time, but would have restricted the system to linear movement. Without a clearly superior picking motion available in the literature for plums, it is not possible to determine which degrees of freedom are required, so an articulated arm is chosen for development flexibility.

A UR5 CB2 articulated robot arm is used for testing. This is a 6 DoF arm with a 5kg payload, being a collaborative robot (CoBot), farm workers are also able to safely harvest fruit around the robot while it operates. Commercial deployment of such a harvesting system would use a cartesian gantry matched to the required picking motion, so arm reach and speed are not major concerns for the prototype harvester. However, basic speed tests were carried out to determine the maximum picking rate of the UR5. The arm was tested on the lab setup described in Section 6.4, and run at 100% of its factory default speed, this can be increased but makes collisions less safe. Including all system components results in a 12s per fruit pick rate, or 300 per hour. Waiting for UR5 motion to complete is 75% of the total time, so actuation

speed is the dominant factor in the picking rate. Moving to a cartesian system can reduce this.

While the pick rate required for viable commercial operation is determined by a raft of factors, our informal discussions with plum growers indicated a pick rate one quarter that of a human would be economic for them. During field trials a mean pick rate of around 4600 plums per hour was measured for experienced human pickers, the current development system is an order of magnitude slower, but can operate 24/7. Picking speed is a critical performance metric for commercial harvesting systems, but is less of a focus than collisions, and the picking success rates, for this initial prototype. Beyond actuation speed, other considerations for agricultural robotics include water and dust ingress, all of which can be better met by linear motion systems.

Control of the UR5 arm occurs using joint wise proportional-integral-derivative (PID) loops running on the standard UR controller box, which also supports basic motion planning. This hosts a full Linux instance and is needed for the computationally demanding kinematics solutions required by articulated arms. Using a highly restricted picking RoI largely eliminates singularities, and is required because the approach motion controller, described below, is unable to perform singularity checking when operating with the CB2 model arm. Later models implement this functionality.

## 5.4 Motion & Control

Effective harvesting requires accurately placing and closing the gripper, then applying the correct motion to detach the fruit without causing damage, these two phases are approach and retraction respectively. While many different picking motions have been proposed for other fruit, no performance studies for plums exist. Human pickers were observed to use a twisting motion about two plum axes to detach the fruit, which was confirmed as the most effective approach by the farm manager. This strategy, shown in Figure 5.6, called *complex motion* was tested, along with a simple straight pull for comparison. Testing the parallel gripper with complex motion resulted in many collisions, so only straight trajectories are trialled for this.

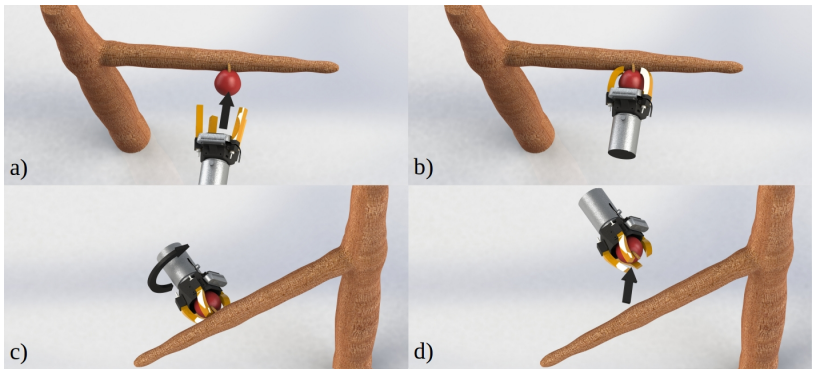


**Figure 5.6** – The complex picking motion consisting of (a) the straight approach, (b) gripper actuation, (c) rotation, and (d) angled retraction.

Moving through the approach motion region of Figure 5.5 will cause contact with soft obstacles and occasionally rigid ones. To minimise the volume swept by the gripper, and to assess the viability of a cartesian gantry system, the goal approach trajectory in this region is constrained to be linear and perpendicular to the trellis. This goal trajectory should terminate at the fruit position estimated using the EKF, however that estimate may be inaccurate and fruit often move as surrounding branches are contacted by the gripper. Feedback from the embedded wide angle camera can be used to update the pose estimate online. So both a direct approach controller using only the EKF information, and a feedback controller using the wide angle camera were developed.

The *direct servo controller* uses the EKF pose estimate without updates from the embedded camera. End effector motion commands from the direct servo controller are sent using the UR script interface. The UR5 controller then performs a differential IK step to calculate instantaneous joint velocities to achieve the commanded motion. This uses the robot Jacobian directly, with no singularity or collision checking performed. A velocity vector perpendicular to the trellis is sent until the desired movement distance is reached.

The IBVS controller runs the object detector model from Section 4.1 on each wide angle camera frame to perform image based visual servoing control of the end effector. This can respond to fruit motion during harvesting and correct for bad initial pose es-

timates, but may become obscured or attempt to reach the wrong fruit. Algorithm 5.1 describes this controller, where $u_{BBox}, v_{BBox}$ is the centroid of the object bounding box. The grip point in the image frame $(u_{target}, v_{target})$ is experimentally calibrated at the goal fruit radius, and the pixel space gains $G_u, G_v$ are manually tuned. Target association is done by finding the detection nearest the image frame centre, with the embedded camera aligned to the EKF position estimate when the IBVS controller is initialised. Each time the lost tracking counter increases, the previous motion command is also re-run.

**Algorithm 5.1:** IBVS Approach Controller

```
while new image frame do
    if time > max time then
        Return status = time limit
    if distance > max distance then
        Return status = distance limit
    detections ← runDetector(image);
    if no detections then
        lost tracking counter++;
        if lost tracking counter > 15 then
            Return status = tracking lost
    else
        lost tracking counter = 0;
        Bbox = min_{Bbox∈detections} |(u_target − u_Bbox)| + |(v_target − v_Bbox)| ;
        if Bbox radius > max radius then
            Return status = success
        sendNewControlSignal(Bbox);
```

In Algorithm 5.1 the *sendNewControlSignal* function calculates an error term in pixel space and applies controller gains to this

$$u_{vel} = G_u(u_{Bbox} - u_{target}) \tag{5.1}$$
$$v_{vel} = G_v(v_{Bbox} - v_{target})$$

The z axis preset velocity towards the fruit $ee_{vel}$, is calculated and used to construct the end effector velocity vector $\vec{V}$ as

$$d_{vel} = ee_{vel} - max(|(u_{vel}, v_{vel})|) \tag{5.2}$$

$$\vec{V} = \begin{bmatrix} u_{vel} \\ v_{vel} \\ d_{vel} \end{bmatrix}$$

this end effector velocity is scaled and sent to the UR5 controller box

$$\vec{V}_{cmd} = ||\vec{V}||_2 \times ee_{vel} \tag{5.3}$$

Both controllers account for joint limits by stopping motion if any joint comes within 2 degrees of its limit. If any joint exceeds the software set limit, the previous end effector motion command is inverted and sent to bring that joint back within range. Stopping with a *success* condition occurs if the largest object detection exceeds a set radius, with such a wide angle camera, this is a reliable indicator of a fruit within the gripper cup.

Singularities, unreliable detections and obscuration are issues for the IBVS controller. Moving through singularities while commanding an end effector velocity results in infinite joint speed commands, this e-stops the robot which can be detected and reset in software. A similar process occurs for collisions and both cause the IBVS controller to return with an error state. Unreliable detections or obscurations can mean the target is lost for many frames, also causing an IBVS error. Where one target is lost but other fruit are in frame, the controller will switch to tracking those. Oscillating behaviour can result from this, where the controller switches between two targets multiple times. During preliminary lab testing this was observed only a handful of times, but is more likely if detector performance is low. By re-sending the previous command for frames with no detections, the camera is often able to push past obscuring leaves to regain tracking on the target.

This form of controller is non-linear due to target association and Equation 5.2, but was found to be stable and effective in practice across wide ranges of $G_u, G_v$. Both

parameters are also dependent on the framerate, and hence the controller update rate, of the embedded camera and object detector. Safety is the primary concern when setting $ee_{vel}$ and encountering singularities can be dangerous regardless of end effector speed.

## 5.5 Picking State Machine

When performing autonomous harvesting, the system operation can be described by a state machine, shown in Figure 5.7. Actuation goal poses and motion planner settings are the main output of this. It is able to switch between RRTConnect for global planning or one of the approach controllers for moving through the obstacle planes. Actuation goals consist of 5 end effector poses which are defined at all times, these are a compact home position, a pose above the fruit basket, a look position, an approach pose offset from the target fruit by 150mm to place it outside the right-most obstacle plane and the current target fruit position from the EKF. Operators are able to manually actuate the gripper, set or recall the actuation goals, or run autonomous harvesting. The home position is used for transporting the arm and when initialising the system. Between picking motions the system returns to the look pose, which is used to image a large area of trellis while being close to the RoI to minimise actuation time.

Autonomous picking currently occurs once per mobile platform base position, so all fruit in the RoI are harvested, the platform is then moved forwards and the process repeats. Picking while moving at slow speeds should be possible with wheel encoders fitted. When picking starts, the EKF state vector is cached and each fruit from this is added to the global planning frame. The state machine then attempts to pick these in order of decreasing height. More advanced ordering approaches, such as travelling-salesperson formulations, are not required for a static fruit basket. The optimal starting fruit is that which falls furthest on the vector connecting the drop to the home position, other fruit orderings do impact overall picking time. Identifying fruit which prevent access to others, and must be picked first, would improve overall

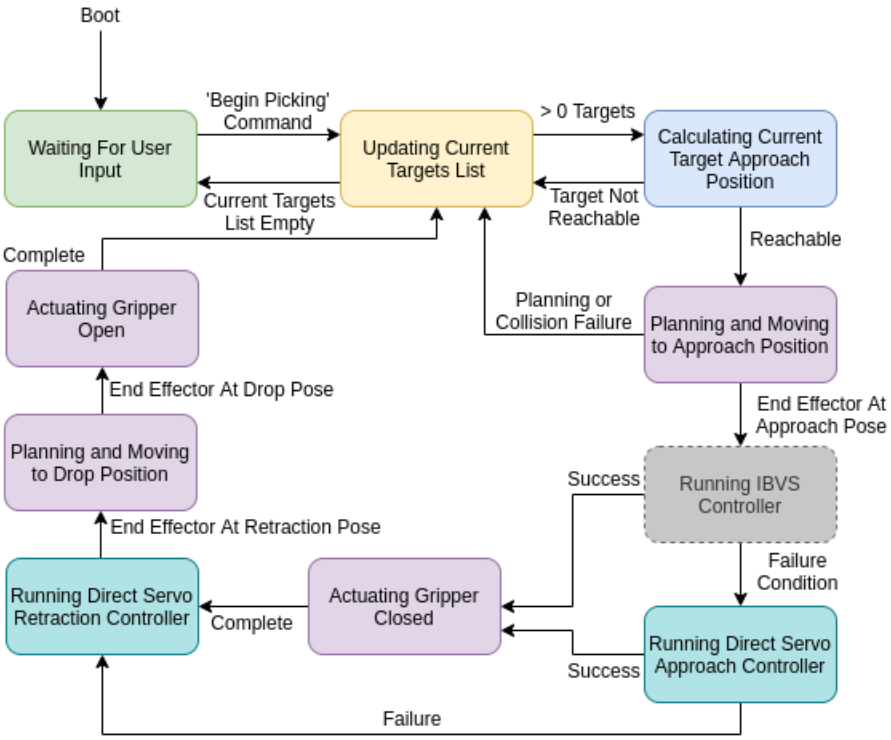


**Figure 5.7** – The state machine used to control picking. Not shown are the user input states, including recording or recalling key arm positions.

performance and is something to be examined in future.

When moving past the fast motion obstacle plane, the IBVS controller takes over, if any error condition occurs the system switches to the direct servo controller. If that also fails the state machine moves to the next fruit. Upon approach controller success the gripper actuates and one of the retraction motions is applied. For the parallel gripper, force and torque feedback indicate if nothing has been grasped. If that occurs the IBVS approach method is rerun once. This controller, along with the gripper design, picking motion and mixed obstacle planning approach, are tested during the field trial described in the following chapter.

# Chapter 6

# System Implementation & Field Evaluation

This chapter details implementation and field testing of the full system prototype including all functional modules previously described. Figures in this chapter are adapted from Brown and Sukkarieh (2021), and some additional experimental method details are available there. The chosen environmental representation of primitive shapes and collision meshes is combined with a stereo RGBD, and monocular RGB, camera to sense target fruit. Object detection is run on RGB frames and the resulting bounding boxes are combined with depth information to perform fruit pose estimation. These poses are Kalman filtered and transformed into a global frame for persistent tracking.

The picking state machine determines the action sequences, with both IBVS and direct approach controllers used for final gripper positioning. Retraction motion and gripper type are both essential design decisions, so the alternatives presented in Chapter 5 are evaluated using the prototype harvester. System field trials occur on a commercial plum crop in Swan Hill, Victoria. Details of the trellis type used are presented here, along with motion and gripper comparative studies, and overall system performance.

## 6.1 System Hardware

System hardware consists of supporting elements such as power, mobility, and computation, along with robotics specific elements of sensing and actuation. Supporting hardware selection is relatively straightforward, but concern must be paid to the ruggedisation, endurance, size and weight of components. Hardware system design goals were to build a fully self contained development platform with long endurance, modularity, and support for additional types of payloads in the future. Agriculture specific constraints include being able to fit within 2.5m crop rows, access a 3m tall picking window, and turn on a 3.5m row headland. All of these goals were met with the use of a trailer base and onboard generator.

### 6.1.1 Supporting Hardware

To maximise flexibility of the support hardware, a mobile trailer base is used which can be towed by a robot or farm vehicle, simplifying trial logistics. Key hardware components can be seen in Figure 6.1. Commercial off-the-shelf parts are used wherever possible to minimise cost, though the gripper, gripper control unit, and sensor mountings are all custom fabricated. Compute and support components are mounted on the lower level of the trailer, with actuation and sensing on the upper platform.

Modularity of both hardware and software is a focus, so all components are mounted for easy removal using a tray and latch system. Water and dust resistance is provided by a hook and loop secured skirt that encloses the bottom level of the trailer. Components on the platform top, such as the robot arm, are IP54 rated, whereas IP55 would be required for true waterproof operation. An uninterruptible power supply (UPS) provides a 20 minute battery runtime, allowing the generator to be refuelled while the system is active. Trailer motion from arm inertia was a predicted issue but was not seen in practice.

A 3kW petrol generator provides 240V power, with system demands shown in Table 6.1. Compressed air at 60psi is required to actuate the soft gripper design. This

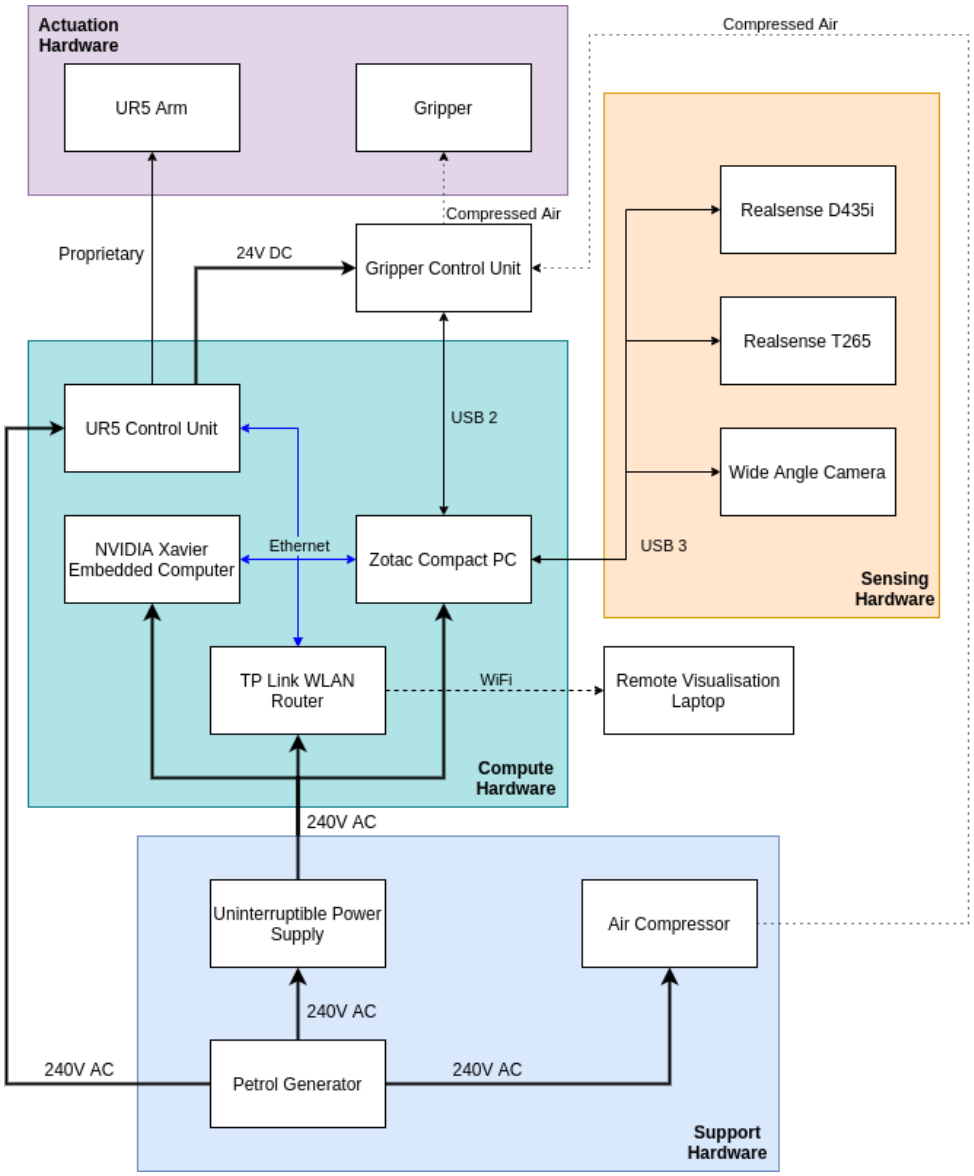


**Figure 6.1** – System hardware modules and connections, grouped by function.

comes from a small compressor with an onboard storage tank, the duty cycle of this is close to 5% resulting in a low average, but high peak power draw.

| Component | Nominal Power (W) | Maximum Power (W) | Input Voltage (V) |
|---|---|---|---|
| Compressor | 50 | 1000 | 240 AC |
| Zotac | 200 | 330 | 240 AC |
| Arm | 150 | 325 | 240 AC |
| Xavier | 40 | 75 | 9-20 DC |
| Router | 10 | 10 | 12 DC |
| Arduino | 1 | 5 | 5 DC |
| D435 | 1 | 2.5 | 5 DC |
| T265 | 2 | 2.5 | 5 DC |
| Wide Angle Camera | 2 | 2 | 5 DC |

**Table 6.1** – System power budget

### 6.1.2 Computing Hardware

Compute and networking are handled by a general purpose PC, embedded deep learning computer, router, gripper control box, and hubs for USB devices.

A Zotac EN72080V mini PC is used as the general purpose computer, this contains an i7 processor, 32GB of RAM, and an NVIDIA RTX2080 GPU. To deploy deep learning models in a power efficient manner, and to reduce the Zotac system load, an NVIDIA Xavier embedded computer is used for deep learning inference.

### 6.1.3 Sensing & Actuation

Three forms of sensing are required, long range fruit detection and localisation, short range detection for feedback control, and platform position tracking. The first is provided by a Realsense D435i RGBD camera mounted above the gripper. This provides both an RGB image for object detection and a depth map to estimate the 3D location of fruit and obstacles. Platform position tracking is done using a Realsense T265 simultaneous localisation and mapping camera mounted to the rear corner of the trailer. Platform localisation is important for tracking targets as they move in and out of frame, additionally it allows for orchard scale data registration for yield mapping and health monitoring. Data streams for the IBVS approach controller are

provided by a wide angle (170° diagonal FoV) eye-in-hand camera in the soft gripper.

Rapidly iterating on multiple gripper designs is an important aspect of system development. So the modular approach was extended to this area by defining a design agnostic interface consisting of a gripper control box and ROS action service. The gripper control box takes standard form ROS action requests and performs low level servo or valve actuation to control gripper motion according to these. Identical physical and electrical input characteristics mean grippers can be quickly swapped in the field. A UR5 CoBot arm is mounted on the trailer top with the soft gripper attached to this. For parallel gripper testing, the end effector and gripper control box are swapped, with positioning done manually.

## 6.2 System Software

The robot operating system (ROS) software is chosen to handle process communication and the launching of compute nodes. This is inherently modular with well defined interfaces, known as ROS topics, services or actions, between processes. Process dependencies match the functional module order of Figure 6.2 and were developed in that order. The perception module, for fruit localisation in a global frame, was first developed as a standalone ROS package. This could then be applied to other tree crop projects for counting, health assessment or yield estimation. The modules following that were then developed and tested in the lab. A simple parallel gripper was the first design used, described in Section 5.1.1, and the system was then taken for testing on a nearby apple crop.

Field experiments use the YoloV3 embedded detector architecture running on the NVIDIA Xavier, with the comparative study in Section 4.1 conducted after the field trials were complete. HSV patch filtering, described in Section 4.3.1, is applied to estimate fruit poses which are stored and filtered within the EKF state vector.

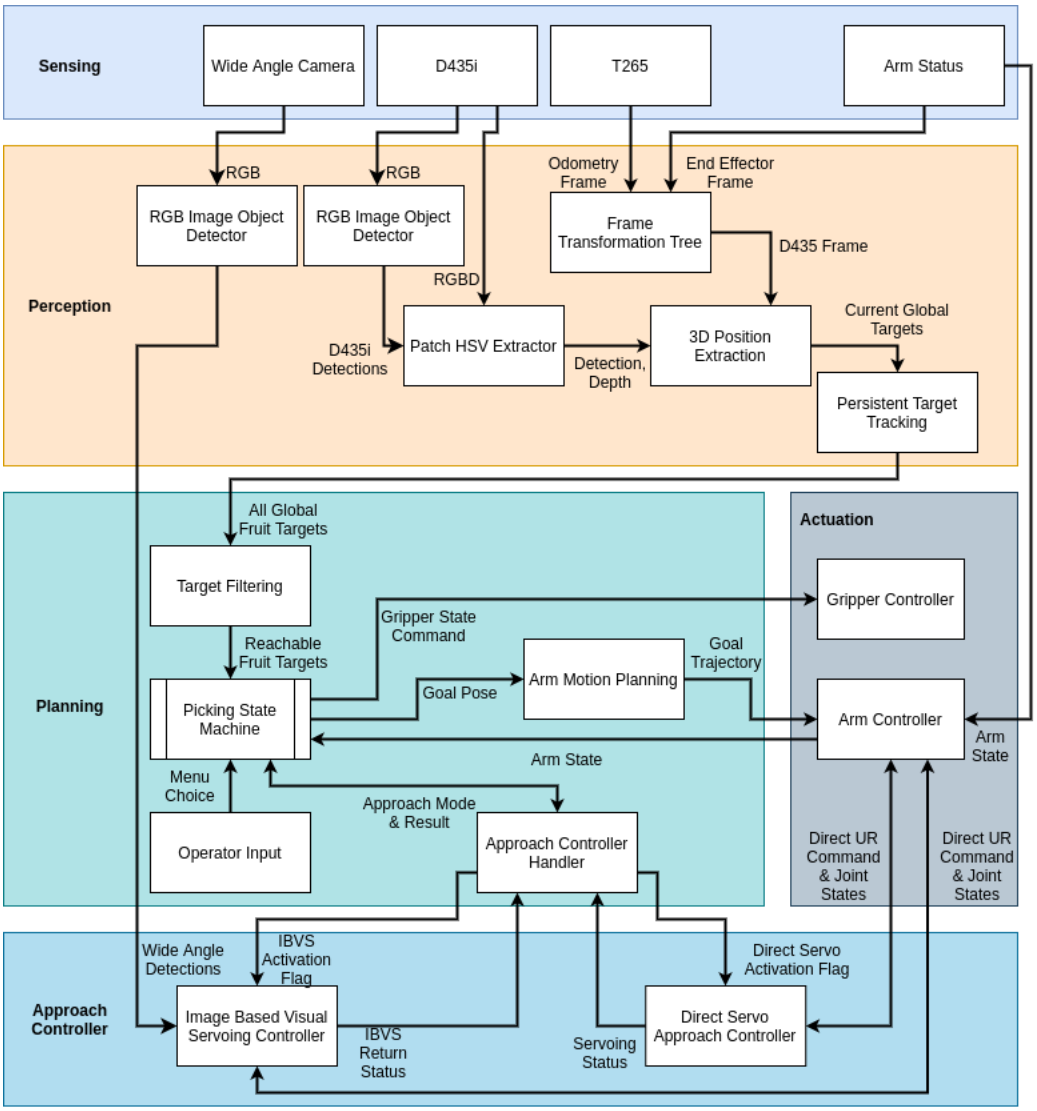


**Figure 6.2** – Software architecture for the as-tested prototype harvesting system. Arrows indicate data flow direction

## 6.3 Field Testing Phases

Following the development of all major prototype components, the full system underwent three phases of testing. First, lab testing was carried out on the trellis simulation described in Section 6.4, to identify and fix system faults. Trials on a locally grown apple crop followed this to provide qualitative data on component performance amid more realistic obstacle and sensing conditions. Finally, a full-scale week long trial on

a commercial plum crop was carried out in Swan Hill, Victoria. One reason for using a three phase trial is the narrow harvesting window present for most tree crops. This limits final testing to a handful of weeks each year for any given plum variety.

During lab testing, many issues were rapidly identified and corrected. Key to this step was the effective visualisation and data flow inspection tools available within ROS. As a development decision, all tunable module parameters were put on the ROS parameter server and coded for online updates. While this added development overhead, it was instrumental during the lab testing phase and saved saved valuable field trial setup time. Previously mentioned issues with approach controller singularities, calculating fruit depth using overlapping leaf depth readings and determination of overall actuation time were resolved at this point.

With the system reliably able to pick fruit in the lab simulated trellis setup, testing moved to local apple crops. These provide a realistic collision environment of hard branches with dense leaves. Unlike the final plum crop, these are grown in semi-flat trellis, with branches allowed to extend up to 1m from the trellis plane. This invalidated the assumptions in Section 5.2 resulting in many problematic collisions and reinforcing the importance of using fruiting wall style trellis. Moving rigid gripper fingers past obstacles, as described in 5.1.2, was identified as required functionality during this testing. Gripper design refinements were the primary outcome of apple testing, with the object detection, localisation, and tracking shown to execute without bugs or significant errors under field conditions.

Trials on a commercial plum crop took place over a week in Swan Hill during normal harvesting operations, as described in Section 6.5. Overall system success rate was determined, and key design choices around the gripper type and picking motion were evaluated. The ‘Late Scheffer’ plum cultivar type was used for testing, grown in the fruiting wall configuration described in Section 6.4. Bunching of fruit was one issue observed for this crop, a common phenomenon driven by inconsistent flower thinning. This complicates both autonomous and manual picking, and reduces fruit quality leading to disease and bruising where fruit are in contact. Feedback from the grower indicated that he expects to largely eliminate bunching through improved

thinning methods the following season.

## 6.4 Trellis Type and Parameters

Success of autonomous harvesting is highly dependent on the cropping conditions encountered. Trellising is the primary component of this and consists of the supporting and training system for trees, typically made from rigid posts and metal wires. A variety of growing systems are used in Australia, but one recent paradigm has been the Simple, Narrow, Accessible, Productive (SNAP) principle, as in Gao et al. (2020). Flat 2D trellis types, also known as fruiting walls, meet this definition and are becoming increasingly popular. These provide excellent access for mechanised tools, which is one of the factors currently driving their adoption among growers. The flat surface, excellent fruit access and easy pruning rules that make these suitable for mechanisation also make fruiting walls an ideal candidate for robotic harvesting. This 2D trellis type is used for development and testing, as seen in Figure 6.3, but other forms such as V or T trellis types are also common.

As with many fruit, plums develop from individual flowers which grow in clusters. Left un-thinned, the large number of natural flowers compete for tree resources resulting in smaller and lower quality fruit. Effective flower thinning is identified as the most important factor in growing high-quality fruit by Looney (1993). Thinning can be done chemically or physically and is often a blanket treatment followed by hand thinning. Typical regimes aim for one to three fruit per flower cluster, and hand thinning allows the exact fruit position to be chosen from each flower bunch. Thinning strategies can be selected to support autonomous harvesting, such as eliminating fruit behind branches or near trellis posts.

Progressive system development and testing required a lab simulation for early trials. This was constructed as a fruiting wall using synthetic trees with steel washers fixed to these at various locations. This allowed multiple synthetic fruit types to be fitted with magnets for easy reconfiguration and resetting after tests. Synthetic oranges, apples, lemons and plums were tested using this magnet attachment method in order

**Figure 6.3** – A fruiting wall trellis section with immature plum trees, clearly showing how these are trained into 2 vertical trunks which are fixed to the trellis wires and to reinforcing bamboo poles, which are later removed. Fruit grow directly on the trunks and small horizontal branches which are pruned annually into the 2D plane of the fruiting wall.

to explore different target sizes and appearances. Ideal detachment force was similar to that of a ripe plum, but varied as the magnet did not always sit flat against the washers where branches or leaves were in contact with it. Tree bases were weighted in place, allowing them to move upon significant collisions. Likewise, the trellis wires were held under spring tension so that collisions would trigger the arm emergency stop but not damage the gripper or trellis setup. This flexibility allowed problematic collisions to be identified during testing without them causing damage.

Shape and visual properties were well captured by the simulated trellis through the use of photorealistic fake trees and fruit designed for interior decoration. These allowed perception components to be tested, though this simulated trellis does have several limitations. The synthetic trees are cast from a semi-flexible plastic with an internal metal support structure. This was more flexible than true plum branches and much less abrasive, both such issues were encountered in field trials. Flexibility also led to large fruit and branch movements when approaching or picking fruit. Indoor lighting is also more regular, while rain, wind, or dust are hard to simulate.

## 6.5 Plum Harvesting Experiments

Three experiments are conducted using the commercial plum crop, these are an overall system evaluation, plus gripper type, and picking motion studies. The gripper type experiment compares the two completed grippers from Section 5.1 using simple motion. While the picking motion experiment compares the two motion types from Section 5.4. Overall system evaluation is done using the best found combination of gripper and motion, which is the soft gripper and complex motion.

Key overall metrics are pick attempt success rate, collision rate and hardware failure rate. For several fruiting wall trellis sections all fruit falling between the middle trellis wires, corresponding to the RoI height, were left untouched by human pickers. For each trailer position along the crop row, all fruit detections within the RoI were attempted. The platform was then moved forward until a 100mm overlap with the previous RoI remained, ensuring all fruit were attempted. Targets appearing in multiple RoI overlaps were manually excluded so all targets were only attempted once. No modifications are made to the crop and no fruit, including those behind trellis wires or trunks, are excluded from picking attempts. Parameter tuning, which includes the fixed pose locations, trellis obstacle plane distances and detector confidence threshold, occurred over the first 4 days of the week long trial. Object detector training, and qualitative assessment also took place during this time, under conditions of sun, wind, rain, and darkness. Quantitative testing then occurred over two days. A total of 64 pick attempts were carried out with the nominal configuration of the soft gripper with complex motion. The Section 5.4 simple picking motion was also tested on 20 soft gripper pick attempts. Comparison of the soft and parallel grippers took place by manually positioning the latter over target fruit in either a vertical or horizontal orientation, then closing it and applying the simple retraction motion. Complex picking motion with the parallel gripper resulted in damaging collisions for almost every attempt and was too dangerous to further evaluate without risking arm or tree damage.

While lab testing and initial field trials indicated that the IBVS controller was es-

sential to correcting for picking induced fruit movement, tests on plums during the first field trial day saw almost zero fruit movement. The close proximity of plums to branches and more rigid trellis structure reduced fruit motion below the gripper position tolerance. While the IBVS approach controller is quite robust, with almost no fruit movement any IBVS errors outweigh the benefits of this feedback controller. Instead, the direct servo approach controller was used for all plum harvesting attempts.

## 6.6 Results

Pick success rate for the best system configuration was 42%, below that required for commercial viability but reasonable performance for an initial prototype. Comparison with existing literature results is difficult, as the majority of papers report figures for modified crops, or where only easy fruit are attempted. Some system components were clearly identified for improvement. The parallel gripper was ineffective due to collisions and failed picks, while the harvesting motion chosen is critical to success. Damaging collisions were eliminated, with no damage observed when using multi-plane planning and the soft gripper, but soft actuator longevity is one challenge.

The lower than expected recall from YoloV3 led to the post-trial analysis of object detector architectures, presented in Section 4.1. Use of embedded software components may have impacted this, so Section 4.1 uses a non-embedded YoloV3 version. During the trial a basic HSV thresholding object detector was tested as an alternative to YoloV3, but proved unsuitable. Results for this are in Table 6.2 where a positive detection is assessed using a 0.5 IOU bounding box overlap.

The pre-trial testing described in Section 6.3 showed the importance of dealing with collisions and the effectiveness of soft components for picking success. Numerous emergency stops were triggered by contact between branches and the rigid parallel gripper fingers. These failures alone are sufficient to prevent this design being practically used, but even without collisions the parallel gripper only succeeded 30% of the time. Success for this gripper, as for the soft gripper, is defined as detaching a

| | HSV Day | YoloV3 Day | HSV Night | YoloV3 Night |
|---|---|---|---|---|
| True Positives | 57 | 73 | 30 | 13 |
| False Positives | 14 | 0 | 6 | 0 |
| False Negatives | 170 | 154 | 40 | 57 |
| Mis-Separation | 24 | 2 | 4 | 1 |
| Recall | 25.1% | 31.2% | 42.9% | 18.6% |
| Precision | 80.3% | 100% | 83.3% | 100% |
| Bad Box Rate | 42.1% | 2.7% | 13.3% | 7.7% |

**Table 6.2** – Embedded YoloV3 and HSV object detector performance during initial field trial evaluations

fruit without dropping it. As shown in Table 6.3 the rigid gripper orientation is important with vertical, fingers above and below the fruit, outperforming horizontal by a factor of 3. Parallel gripper failures typically occurred due to an obstacle coming between the finger and fruit. Most obstacles exist on the sides of fruit, instead of above or below them, explaining the importance of orientation for this gripper. Soft fingers, which can deform over obstacles and re-contact the fruit, do not have the same dependence on orientation. One limitation of the gripper design study is the small number of tests that could be carried out. For the rigid gripper, much time was lost to collisions, meaning fewer assessable attempts could be completed in the time available.

| Manipulator Type | Successes | Failures | Success Rate |
|---|---|---|---|
| Soft Gripper Simple Motion | 4 | 16 | 20.0% |
| Soft Gripper Complex Motion | 27 | 37 | 42.2% |
| Horizontal Parallel Gripper | 2 | 8 | 10.0% |
| Vertical Parallel Gripper | 6 | 7 | 30.0% |

**Table 6.3** – Picking success rate by gripper and motion type

Soft gripper performance was more than doubled when moving from simple to complex picking motion. One reason for this is the mechanics of fruit detachment. Mature

plum fruit naturally release at the abscission layer when ripe and will break at this layer if shear force is applied. Straight tensile force does not cause the same abscission detachment and requires much more total force to remove the fruit. It also risks stem pull-out which can damage the fruit by breaking the skin allowing diseases and contamination in. Complex picking motion results in much more shear than tensile force and mimics the behaviour of human pickers, leading to more detachments at the abscission layer with lower net force. Low individual finger force is one limitation of the present soft gripper design, meaning picks were rarely successful unless all fingers could be in contact with the fruit. An additional benefit of complex motion is the twisting action allows the fingers to fall into a more closed stable mode where the fruit is better in contact, leading to more force transmitted. This effect can be seen in the bottom row of Figure 5.3.

Reasons for soft gripper failures are Tabulated in 6.4. Bad positioning is assigned when a more accurate gripper pose is expected to have enabled a pick success, such as by having more fingers in contact or avoiding nearby obstacles. Knocked off target failures are often a specific type of bad positioning where a finger contacted the target fruit centre and knocked it off before grasping it. Gripper failures were unfortunately common with this finger design, which would be improved with more robust materials in future iterations. However, it was observed during testing that the pneumatic finger actuators have mostly graceful failure modes when ruptured. In most cases, small leaks developed in the finger end caps leading to a partial pressure loss in the common air loop. The supply rate of the compressor and sufficient diameter tubing meant that other fingers could still operate at the lower pressure with only a small drop in holding force. For the damaged finger, the balance of air pressure determined the actuation force, such that small leaks would reach pressure equilibrium while still working as a gripper, but without sufficient air pressure to develop further leaks in that finger. So the pneumatic system can operate for short periods with only minor functionality loss when damaged. Short term operation of damaged fingers, with reduced pressure, is thus possible. Many of these failure modes are mutual and assigned causes will be imperfect. For example, additional gripping force can make some poorly positioned

picks successful which otherwise are failures.

| Outcome | Straight | Angled | Straight Percentage | Angled Percentage |
|---|---|---|---|---|
| Success | 4 | 27 | 20.0% | 42.2% |
| Grip Force Failure | 7 | 5 | 35.0% | 7.8% |
| Bad Positioning Failure | 6 | 9 | 30.0% | 14.0% |
| Knocked Off Target Failure | 1 | 9 | 5.0% | 14.0% |
| Gripper Failure | 0 | 2 | 0.0% | 3.1% |
| Other Failure | 2 | 12 | 10.0% | 18.8% |
| Total | 20 | 64 | | |

**Table 6.4** – Harvesting failure modes by picking motion

Many of the 'other' types of failures occurred with fruit in positions impossible to properly reach using cartesian motions, including behind trellis wires or tree trunks. One downside of the complex motion was increased fruit loss when grasps failed. Failure of the straight detachment movement would often leave the target fruit on the tree where human pickers could still collect it, but complex motion failures would usually detach the fruit, dropping it to the ground. No fruit bruising or skin tears were observed to be caused by either gripper during successful picks. Additionally, no emergency stops were caused by soft gripper collisions, indicating that the assumptions in Section 5.2 are correct.

## 6.7 Field Trial Discussion

System performance in the commercial plum crop field trial was a long way from commercial viability, but clearly identified challenges and opportunities for improvement. The key assumptions around mixed obstacle operation using planning planes and soft robotics were demonstrated to be accurate by not having any damaging collisions when using this configuration. Modular system design allowed the gripper hardware and motion type to be easily changed in the field. Additionally, valuable

data on object picking motions for plums and fruit localisation methods was gathered.

One observation was the importance of frequent testing on realistic crop scenarios. In both the lab and apple trials, an IBVS approach controller was essential to correct for fruit movement during picking. When testing on plums, fruit motion was minimal but dense leaf cover made the visual servoing approach counterproductive. This sort of lesson can only be determined by realistic field testing, which is complicated by the narrow picking window available each year and variety of trellis systems in use.

The EKF based continuous target tracking proved instrumental in accounting for poor object detector performance, which is difficult to overcome in highly obscured cultivars. By integrating this continuous tracking system, other properties such as fruit health, could be estimated in future by using multiple sensor views. Waterproofing was a platform requirement that was unexpectedly tested, with several trial days lost to heavy rain. The present hardware is only designed for light rain exposure which it was able to operate in. While designing physically waterproof platforms and sensors is not difficult, the sensing performance of that hardware in heavy rain, along with the impact of altered friction properties on gripping, remains to be tested. Harvesting at night was tested using artificial light, with no observed drop in effectiveness. Picking often occurs overnight with human workers, to take advantage of lower temperatures and calmer wind. During very hot days, harvesting is stopped as fruit bruise from the pressure of being piled in picking bins. Use of artificial illumination, with high power lights or strobes, can be applied during day operations but at the cost of additional hardware, power use and complexity.

Offloading object detector inference work to the embedded computer is beneficial for power use and loop time stability, which in turn is important for good EKF functioning. Local network load induced by this was reasonable at 40 MB/s with few efficiency optimisations within the tested code. Off the shelf computing and open source software messaging frameworks, such as ROS, are now sufficient for both developing and deploying agricultural robotics platforms unless power efficiency is of particular concern. Performance of the YoloV3 model was a limitation and may be linked to the embedded implementation used, as indicated by Section 4.2.

Study of the two picking motions highlighted the importance of incorporating existing harvesting and crop biomechanics knowledge. Even small changes to picking motion took the soft gripper from a success rate of 20% to 42%. However, this also resulted in longer pick time and increased abrasion of the soft fingers. Failure of these fingers was a persistent issue, with 2 complete failures and 2 minor ones occurring over roughly 300 total trial picks. Each time a finger is compressed between the rigid gripper base and wires or branches, minor damage results to the external tensioning fabric and internal pneumatic bladder. These finger prototypes are a first iteration and can be improved with more robust material selection. The parallel gripper was unusable due to rigid component collisions causing frequent emergency stops. These stops are additional to the failures reported in Table 6.3, meaning performance of this gripper was poor even when collisions did not occur.

### 6.7.1 Plum Specific Observations

Harvesting conditions specific to plums, such as detachment force, close proximity of fruit to their associated branches, and tendency for bunching, were all encountered, highlighting the importance of testing autonomous systems on a wide range of crops.

Knocking surrounding fruit off when picking from a cluster is a problem. More regular and evenly spaced flower thinning is an effective technique to reduce plum clustering. So future efforts at picking clustered fruit should be assessed within ideally thinned crops, and thinning alone may be enough to address the bulk of this problem. Pruning strategies likewise influence the position of branch obstacles, flower clusters and obscuring leaves. Careful pruning strategies may be able to increase the separation between fruit and nearby obstacles, making harvesting an easier problem

During qualitative apple testing the final approach controller and gripper dimensions were observed to be well suited to apples. So the soft gripper design was informed by the fruit size, stem length, and separation of apples from branches, all of which were reduced in the plum crop. This resulted in a slightly oversized gripper which was partially remedied by the soft skirt attachment. Fruit proximity to obstacles

significantly increased collisions while the stiffer and more lignified plum branches made these more problematic. The same two conditions also made the IBVS approach controller unnecessary for this plum cultivar. Detection of problematic fruit to avoid pick attempts for, such as those behind branches or trellis wires, is a key future functionality for the system. This could reduce the fruit lost, time wasted, and gripper damage that results from attempting picks on unsuitable targets. By only approaching fruit from the front, some targets may become harder to pick. The impact of this in plum crops is unknown, but should be explored.

Fruiting wall forms of trellis appear well suited to autonomous harvesting of tree crops. Assumptions of soft and hard obstacle planes held for this unmodified crop undergoing normal commercial harvesting using human labour. Flower and fruit thinning directly impact the fruit distribution, clumping and proximity to branches or trellis wires. This makes it an essential consideration when assessing autonomous harvest success, albeit a difficult to measure and rarely reported one. To address this, raw sensor data for this trial is made available.[12] Both Australian and international growers have expressed a willingness to alter growing systems for more effective mechanisation, and eventually, automation. Part of this task falls to the robotics community, who must identify, test, and recommend suitable trellising systems for automated planting, tending and harvesting of tree crops.

Cultural preference means some plum markets require stems left on, while others remove them. The commercial crop tested on had no requirement for stem length, but the preservation of stems during harvest for certain markets is a challenging robotics task requiring either stem cutting or reliable detachment with the stem intact.

## 6.8 System Design Assessment & Discussion

Aside from gripper design, detector architecture and harvesting motion which are all explicitly tested, this section provides an analysis of other design decisions following

[12]http://data.acfr.usyd.edu.au/Agriculture/PlumHarvesting/

the field trial experiences. Environmental representation and partitioning, sensing, control, and estimation all had impacts on performance. Some broader observations regarding robotic G&M in agriculture are also made.

Considerable development and testing effort went into perception elements for the system, including object detector selection, filter tunining, target localisation and hardware calibration. By using modular systems, the slow to develop perception components could be more easily reused between crop types. This would reduce prototyping cost and time, while allowing for more rigorous assessment of gripper and motion design choices for various crop types. One disadvantage of eye-on-hand sensing is the dependence between actuation and perception, which makes simulating actions difficult. For example, unit tests of the IBVS controller are not possible without some form of perception running.

Combining primitive shapes with collision meshes proved to be an effective ER. No collisions between modelled components, including the trailer, arm, gripper and trellis wires occurred. All information required for the basic grasp planner, namely the fruit centroid locations, was easily and efficiently stored, visualised and updated. The minimal compute requirements for ER updates and storage also allowed more focus to be placed on processor intensive sensor processing and perception. One downside of the current ER is the lack of obstacle representation, some fruit are better accessed from the left or right side and the ER does not currently store the location of branches required to determine this.

Use of a restrictive RoI was effective for eliminating arm singularities in the approach controller joint space, but did make the picking area quite small. The UR5 CB2 model used for development was slightly outdated at the time and the following CB3 model does support singularity avoidance while jogging joints. This would allow a much expanded RoI. For harvesting, once a picking motion is determined a primarily cartesian system can be built to support the required DoF, eliminating the singularities and complex planning associated with articulated arms, while decreasing actuation time.

While the embedded wide angle camera was not used, because the IBVS controller

was ineffective, this sensor was valuable for assessing why picks failed and provides a good view of fruit and finger positions within the gripper. This information is normally obscured, making failure analysis difficult. Depth from stereo vision, as implemented by the D435i, was effective and provided sufficiently accurate position estimates for most fruit despite rain, darkness, direct sunlight, and moving leaves. In future, this depth information can be used to estimate the distance to hard and soft obstacle planes online. Being the first pipeline stage, sensing robustness is essential. Fruit localisation reliability within plum crops is achieved by careful assessment of the perception algorithms used, such as the object detector architecture study, and by using well proven commercial sensing solutions.

Arm and gripper control presented no issues, force feedback from the parallel gripper was beneficial for regulating fingertip pressure and detecting failed grasps. Incorporating sensing into soft robotics components is difficult, but these partially overcome the need for feedback control by conforming to the fruit surface. A very low modulus of elasticity means strain due to mis-positioning or obstacles does not result in significant stress applied to the fruit or finger. Tolerance of the soft gripper to poor positioning offset some errors in the perception and actuation stages, successfully gripping the fruit where rigid grippers are likely to have failed.

Filtering and estimation likewise can offset object detector errors by tracking fruit which were occasionally missed by the detector. While more advanced frameworks exist, the EKF approach to tracking is efficient and easily implemented. Variability in sensor frame rates was not a problem, and the Euclidean distance metric for target association worked well with minor tuning. The optimal value for this parameter is crop dependent, so more robust approaches such as visual feature matching should be investigated to remove that dependency.

Some lessons from agricultural tasks can be applied to robotic G&M in general. One is the slow development that results from weather, seasonality, access, and ergonomic challenges associated with field testing. As robotic platforms are increasingly deployed to unstructured environments, the difficulty of effectively testing, debugging and developing with unreliable internet connections, days lost to rain, screens un-

readable in direct sunlight and similar factors should not be underestimated. Unlike indoor environments, outdoor sensing can never be made entirely robust. There is always the possibility of a stray leaf, mud fleck or insect completely obscuring a sensor. Detecting and intelligently handling these cases remains a challenge which is rarely addressed in indoor robotics.

The results presented here are limited to a single season, and further testing in following harvest seasons, which occur annually for this cultivar, is important. Travel restrictions due to Covid-19 prevented planned field work from taking place following this trial, including follow up experiments on soft fruit harvesting.

Moving from human built to agricultural scenes also changes the form and appearance of objects in sensor data. Most constructed items, from buildings to coffee cups, can be described by a small number of geometric primitives making primitive or mesh representations very efficient and allowing properties such as pose to be easily defined. Trees, plants, and other natural objects follow much more complex, and often fractal, geometries. This makes perfectly scanning and storing a tree using a mesh representation very time and memory intensive. Even simple tasks, such as defining a tree centroid for localisation require careful consideration, the centre coordinate of bounding box extents will change rapidly as branches and leaves grow or fall seasonally, the lower $Z$ extent could be the ground plane or base of the root system. Wide variation in objects, by the law of large numbers, leads to truly Gaussian distributions for many more scene properties than built environments. This provides mathematical justification for the application of techniques which make Gaussian assumptions, which is common in robotics perception and navigation algorithms.

Harvesting of soft skinned fruit varieties remains an unsolved problem, but our results indicate that soft robotics components can go some way towards addressing the issues of fruit damage and hard obstacle collisions.

# Chapter 7

# Conclusion

While numerous research works on autonomous harvesting exist, it remains an unsolved and challenging problem. Interactions with soft fruit present additional complexities atop the challenges of agricultural robotics, with additional difficulties not seen in their harder skinned counterparts. This thesis seeks to fill gaps in robotics knowledge relating to perception, localisation, control and hardware design for soft fruit harvesting. Specifically, object detector architectures, gripper designs and motion selection for soft skinned fruit harvesting are examined. It is a first step in this direction, with many more understandings around system design decisions for soft fruit interaction yet to be learned.

In this work, an architecture is developed for testing algorithmic and software components on the problem of soft fruit harvesting. This is applied to plums using a prototype system, and is the first robotic harvester tested on that fruit type. The modular and flexible development platform makes use of soft robotics components, various perception techniques, and multi-stage fruit pose estimation within persistent tracking & filtering frameworks. Detailed studies of module improvements are carried out to support key system functions.

Following the analysis in Chapter 3, an environmental representation using meshes and primitive shapes is chosen. This simple and efficient composite ER integrates with three primary sensing cameras; a 3D camera for target localisation, a wide angle

embedded camera for final approach control, and a platform tracking SLAM camera. Stochastic regularisation techniques are applied to shape completion networks for efficient stochastic object sample generation, allowing for grasp performance marginalisation over multiple object reconstruction hypotheses. However, this extends planning time and is not expected to effectively reconstruct the complex and highly obscured organic scenes observed during harvesting. Using a single range measurement is shown to improve monocular depth inference in scenes with simulated ambiguous global scale, but transfer learning of depth map inference from urban to agricultural scenes will required additional datasets.

Fruit localisation is examined, beginning with object detection, then pose estimation, filtering, and tracking. Object detector architectures for eye-in-hand harvesting data are studied in detail. Modern detector architectures showed surprising results, with performance rankings on a new eye-in-hand picking dataset not reflecting published rankings on large computer vision datasets, such as COCO. Fusion of depth information allowed for minor detector performance improvement, with data augmentation being more beneficial.

The autocovariance least squares method is applied to position based visual servoing to improve noise covariance matrix estimates in both EKF and MHE filtering. Significant improvements were seen over grid search tuned filters in both simulated and lab experiments. Using more accurate estimators for the ALS input data further improved performance, though computational complexity restricts this to offline use. Separately, non-greedy solutions to active perception planning are applied to bearings-only fruit localisation using a 2D camera sensor and a gripper geometry informed cost function. Optimally selecting sensor trajectories reduced the most important dimension of position estimate error by 99% and covariance by 75% compared to a straight approach camera path.

Chapter 5 considers the grasping & manipulation challenges which are unique to agriculture, such as mixed obstacles, fruit movement, and fragile targets. These are addressed through careful trellis assumptions and planning constraints, online feedback control for picking, and soft robotics gripper components. An IBVS controller

is developed for final approach positioning feedback. This proved to be essential for effective picking during lab tests, but was not required on real plums due to their short stems and close proximity to branches. An articulated CoBot arm is chosen for safe operation around farm staff while allowing for motion flexibility, and a maximum picking rate of 300 fruit per hour is estimated for this. Design considerations suggest the use of a cartesian manipulator, once the required degrees of freedom for picking motions are known.

Three stages of platform trials are carried out, culminating in a week long test of harvesting an unmodified commercial plum crop in Swan Hill, Victoria. Specific experiments regarding gripper design and picking motion for plums were conducted as part of this trial. Coupling soft robotics components with constrained motion planning and careful assumptions around hard and soft obstacle planes eliminated damaging collisions. Such an approach is only possible with flat trellis structures, such as the fruiting wall type. Tests using a rigid parallel gripper caused many collision-induced emergency stops and was also less successful at picking fruit. Twisting and angular motion during gripper retraction significantly improved the pick success rate with qualitatively different performance to the longer stemmed apple crops seen in system pre-trials. Observations such as these highlight the importance of testing harvesters on less common crop types, while developing highly robust modules which remain unchanged between crops, such as target filtering and tracking.

Overall performance remains a long way from commercial viability and the number of pick attempts is an experimental limitation. This was driven by the small time window for testing on ripe fruit, which makes extensively testing and iterating over design decisions difficult. Covid-19 induced restrictions also limited further field trials. Despite this, each system component performed its intended function and careful design considerations resulted in hardware and algorithms able to harvest soft fruit. Key development and deployment lessons include the importance of parameter visualisation and tuning tools, appropriate trellis structure choice and persistent target filtering to correct for detection errors. Commercial components provided robust and cheap solutions for compute, power and some forms of sensing. Testing on soft skinned

plums highlighted the benefits of soft robotics gripper components for avoiding fruit damage and problematic obstacle collisions.

Broader questions remain to be answered regarding overall farm integration with autonomous harvesters. One such issue is what percentage of fruit must be autonomously harvested before combined efforts between human and robotic pickers are no longer required, and what form this cooperation takes. The social implications of displacing large numbers of vulnerable seasonal worker jobs should not be forgotten.

## 7.1 Contributions

An improved understanding of autonomous soft fruit harvesting is the main contribution of this thesis, explored through the development, testing, and analysis of a prototype plum harvester. Lessons around soft robotics components, mixed obstacle planning, target filtering, active perception for harvesting and picking motion are identified, which may also be applied to other less common tree crop types. Additionally, this thesis has contributed theoretical improvements and experimental validations of core grasping and manipulation functional components used in autonomous picking. These include

- EKF tuning process improvements for IBVS allowing for more accurate filter estimates of fruit position to be developed in future.
- A detailed examination of object detection architectures for camera-in-hand plum harvesting, including a new dataset and testing of depth fusion methods for this application. This allows for more accurate fruit detection for harvesting.
- A new technique leveraging a single range measurement to improve monocular depth map prediction for ambiguous scenes, allowing this technique to be better applied to scenes where scale ambiguity may be present.

- Improved stochastic object representation sampling methods, allowing for better reconstructions of unseen object regions when reconstructing these. Along with a method for marginalising grasps over these reconstructions, to select the best grasp plan for partially obscured objects.
- Experimental validation of a soft gripper design and complex motion strategy for plum harvesting.

## 7.2 Future Work

Autonomous harvesting platforms are close to commercial viability for some tree crop types, but many remain unaddressed, and transferring techniques from a crop such as apples to soft skinned plums is non-trivial. Additional factors including trellis type, crop modifications, and the flower thinning regime, make benchmarking performance difficult and are often not reported. Developing and rigorously testing harvesting systems for less common crop types requires ongoing research. Two specific areas of future work are; developing tools for online assessment of which fruit to avoid picking, and improving feedback mechanisms during harvesting.

Not all fruit in a given region are suitable for picking. Many are behind hard obstacles and will lead to collisions if attempted, so should be left for human workers or approached from the opposite side. Identifying these requires the perception of hard obstacles, a challenging research task. Other fruit should not be picked because they are unripe, unhealthy or outside of required size bounds. All of these properties can be determined using the RGBD camera data, although additional sensing modalities for crop health assessment, such as brix meters, may hold value. Extending the system with these capabilities will improve harvest quality while reducing picking time spent on unsuitable fruit.

One beneficial property of plums is the lack of fruit movement during picking, due to short stems and proximity to branches. This allowed open-loop approach controllers to be used, however, better low-level sensing and feedback mechanisms would

still allow more responsive positioning and mid-grasp adjustment. As an illustrative example, tactile embedded sensing in the soft gripper fingers could allow pressure distribution to be monitored during a grasp. Fingers not in proper contact with the fruit could be detected and further actuated to compensate for bad positioning. Feedback control with embedded sensing would reduce the probability of missing or dropping fruit during picking, while eliminating the risk of skin bruising.

Embedded tactile or force sensing for collisions could also reduce the gripper wear caused by these. Currently this occurs using the UR5 arm force sensors, but collision sensing within the fingers would allow the motion to be stopped or adjusted earlier, with less damaging deflection of the soft fingers.

Addressing these areas is one step along the path to efficient and sustainable autonomous farming, where fruit are monitored, managed, cared for, and harvested at an individual level.

# List of References


Anderson, B. D. O. and Moore, J. B. (2012). *Optimal Filtering*. Courier Corporation.

Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43.

Arad, B., Balendonck, J., Barth, R., Ben-Shahar, O., Edan, Y., Hellström, T., Hemming, J., Kurtser, P., Ringdahl, O., Tielen, T., and van Tuijl, B. (2020). Development of a sweet pepper harvesting robot. *Journal of Field Robotics*.

Assa, A. and Janabi-Sharifi, F. (2015). A Kalman filter-based framework for enhanced sensor fusion. *IEEE Sensors Journal*, 15(6):3281–3292.

Atanasov, N., Ny, J. L., Daniilidis, K., and Pappas, G. J. (2014). Information acquisition with sensing robots: Algorithms and error bounds. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6447–6454.

Bac, C. W., Hemming, J., van Tuijl, B. A. J., Barth, R., Wais, E., and van Henten, E. J. (2017). Performance evaluation of a harvesting robot for sweet pepper. *Journal of Field Robotics*, 34(6):1123–1139.

Bac, C. W., van Henten, E. J., Hemming, J., and Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, 31(6):888–911.

Baeten, J., Donné, K., Boedrij, S., Beckers, W., and Claesen, E. (2008). Autonomous fruit picking machine: A robotic apple harvester. In Laugier, C. and Siegwart, R., editors, *Field and Service Robotics: Results of the 6th International Conference*, Springer Tracts in Advanced Robotics, pages 531–539. Springer, Berlin, Heidelberg.

Bargoti, S. and Underwood, J. (2017a). Deep fruit detection in orchards. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3626–3633.

Bargoti, S. and Underwood, J. P. (2017b). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, 34(6):1039–1060.

Barth, R., Hemming, J., and van Henten, E. J. (2016). Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation. *Biosystems Engineering*, 146:71–84.

Bernstein, D. (2009). *Matrix Mathematics*. Princeton.

Bicchi, A. and Kumar, V. (2000). Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 1, pages 348–353 vol.1.

Blais, F. (2004). Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231–244.

Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv:2004.10934 [cs, eess]*.

Bohg, J., Johnson-Roberson, M., León, B., Felip, J., Gratal, X., Bergström, N., Kragic, D., and Morales, A. (2011). Mind the gap - robotic grasping under incomplete observation. In *2011 IEEE International Conference on Robotics and Automation*, pages 686–693.

Borst, C., Fischer, M., and Hirzinger, G. (2004). Grasp planning: How to choose a suitable task wrench space. In *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04*, volume 1, pages 319–325 Vol.1.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Brown, J., Su, D., Kong, H., Sukkarieh, S., and Kerrigan, E. (2019). Improved noise covariance estimation in visual servoing using an autocovariance least-squares approach. *IFAC-PapersOnLine*, 52(22):37–42.

Brown, J., Su, D., Kong, H., Sukkarieh, S., and Kerrigan, E. C. (2020). Improved noise covariance estimation in visual servoing using an autocovariance least-squares approach. *Mechatronics*, 68:102381.

Brown, J. and Sukkarieh, S. (2019). Improving monocular depth prediction in ambiguous scenes using a single range measurement. *IFAC-PapersOnLine*, 52(30):355–360.

Brown, J. and Sukkarieh, S. (2021). Design and evaluation of a modular robotic plum harvesting system utilizing soft components. *Journal of Field Robotics*, 38(2):289–306.

Bulanon, D. M., Burks, T. F., and Alchanatis, V. (2009). Image fusion of visible and thermal images for fruit detection. *Biosystems Engineering*, 103(1):12–22.

Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., and Dollar, A. M. (2015). The YCB object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517.

Canelhas, D. R. (2017). *Truncated Signed Distance Fields Applied to Robotics*. PhD thesis, Örebro University.

Choy, C. B., Xu, D., Gwak, J., Chen, K., and Savarese, S. (2016). 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *European Conference on Computer Vision*, pages 628–644.

Ciocarlie, M., Goldfeder, C., and Allen, P. (2007). Dimensionality reduction for hand-independent dexterous robotic grasping. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3270–3275.

Comba, L., Gay, P., Piccarolo, P., and Aimonino, D. R. (2010). Robotics and automation for crop management: Trends and perspective. *International Conference, Work Safety and Risk Prevention in Agro-Food and Forest Systems, 16-18 September 2010, Ragusa, Italy*, pages 471–478.

Corke, P. I. (1993). Visual control of robot manipulators - A review. In *Visual Servoing*, volume Volume 7 of *World Scientific Series in Robotics and Intelligent Systems*, pages 1–31. WORLD SCIENTIFIC.

Cutkosky, M. R. and Howe, R. D. (1990). Human grasp choice and robotic grasp analysis. In *Dextrous Robot Hands*, pages 5–31. Springer, New York, NY.

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

De-An, Z., Jidong, L., Wei, J., Ying, Z., and Yu, C. (2011). Design and control of an apple harvesting robot. *Biosystems Engineering*, 110(2):112–122.

Diankov, R. (2010). *Automated Construction of Robotic Manipulation Programs*. PhD thesis, Carnegie Mellon University, Robotics Institute.

Diankov, R. and Kuffner, J. (2008). OpenRAVE: A planning architecture for autonomous robotics. Technical report, Carnegie Mellon University.

Dimeas, F., Sako, D. V., Moulianitis, V. C., and Aspragathos, N. A. (2015). Design and fuzzy control of a robotic gripper for efficient strawberry harvesting. *Robotica*, 33(5):1085–1098.

Dragiev, S., Toussaint, M., and Gienger, M. (2013). Uncertainty aware grasping and tactile exploration. In *2013 IEEE International Conference on Robotics and Automation*, pages 113–119.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q. (2019). CenterNet: Keypoint triplets for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6568–6577.

Duník, J., Straka, O., Kost, O., and Havlík, J. (2017). Noise covariance matrices in state-space models: A survey and comparison of estimation methods—Part I. *International Journal of Adaptive Control and Signal Processing*, 31(11):1505–1543.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 2366–2374. Curran Associates, Inc.

Eizicovits, D. and Berman, S. (2014). Efficient sensory-grounded grasp pose quality mapping for gripper design and online grasp planning. *Robotics and Autonomous Systems*, 62(8):1208–1219.

Eizicovits, D., van Tuijl, B., Berman, S., and Edan, Y. (2016). Integration of perception capabilities in gripper design using graspability maps. *Biosystems Engineering*, 146:98–113.

Esehagh Beygi, A., Pirnazari, K., Kamali, M., and Razavi, J. (2014). Physical and mechanical properties of three plum varieties. *Thai journal of agricultural science*, 46.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Feng, J., Zeng, L., and He, L. (2019). Apple fruit recognition algorithm based on multi-spectral dynamic image analysis. *Sensors*, 19(4).

Fernández, R., Montes, H., Surdilovic, J., Surdilovic, D., Gonzalez-De-Santos, P., and Armada, M. (2018). Automatic detection of field-grown cucumbers for robotic harvesting. *IEEE Access*, 6:35512–35527.

Ferrari, C. and Canny, J. (1992). Planning optimal grasps. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2290–2295 vol.3.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059.

Gan, H., Lee, W. S., Alchanatis, V., Ehsani, R., and Schueller, J. K. (2018). Immature green citrus fruit detection using color and thermal images. *Computers and Electronics in Agriculture*, 152:117–125.

Gao, F., Fu, L., Zhang, X., Majeed, Y., Li, R., Karkee, M., and Zhang, Q. (2020). Multi-class fruit-on-plant detection for apple in SNAP system using Faster R-CNN. *Computers and Electronics in Agriculture*, 176:105634.

Ge, M. and Kerrigan, E. C. (2017). Noise covariance identification for time-varying and nonlinear systems. *International Journal of Control*, 90(9):1903–1915.

Gené-Mola, J., Sanz-Cortiella, R., Rosell-Polo, J. R., Morros, J.-R., Ruiz-Hidalgo, J., Vilaplana, V., and Gregorio, E. (2020). Fruit detection and 3D location using instance segmentation neural networks and structure-from-motion photogrammetry. *Computers and Electronics in Agriculture*, 169:105165.

Gené-Mola, J., Vilaplana, V., Rosell-Polo, J. R., Morros, J.-R., Ruiz-Hidalgo, J., and Gregorio, E. (2019). Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities. *Computers and Electronics in Agriculture*, 162:689–698.

Goldfeder, C., Ciocarlie, M., Dang, H., and Allen, P. K. (2009). The Columbia grasp database. In *2009 IEEE International Conference on Robotics and Automation*, pages 1710–1716.

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., and Lewis, K. (2015). Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, 116:8–19.

Grant, M. and Boyd, S. (2013). CVX: Matlab software for disciplined convex programming.

He, L., Wang, G., and Hu, Z. (2018). Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 27(9):4676–4689.

Heikkila, J. and Silven, O. (1997). A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1106–1112.

Hemming, J., Ruizendaal, J., Hofstee, J. W., and van Henten, E. J. (2014). Fruit detectability analysis for different camera positions in sweet-pepper. *Sensors (Basel, Switzerland)*, 14(4):6032–6044.

Hohimer, C. J., Wang, H., Bhusal, S., Miller, J., Mo, C., and Karkee, M. (2019). Design and field evaluation of a robotic apple harvesting system with a 3D-printed soft-robotic end-effector. *Transactions of the ASABE*, 62(2):405–414.

Hsiao, K., Ciocarlie, M., and Brook, P. (2011). Bayesian grasp planning. In *2011 IEEE International Conference on Robotics and Automation*.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., and Murphy, K. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. *arXiv:1611.10012 [cs]*.

Janabi-Sharifi, F. and Marey, M. (2010). A Kalman-filter-based method for pose estimation in visual servoing. *IEEE Transactions on Robotics*, 26(5):939–947.

Kamilaris, A. and Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90.

Kang, H. and Chen, C. (2019). Fruit detection and segmentation for apple harvesting using visual sensor in orchards. *Sensors*, 19(20):4599.

Kang, H. and Chen, C. (2020). Fruit detection, segmentation and 3D visualisation of environments in apple orchards. *Computers and Electronics in Agriculture*, 171:105302.

Kapach, K., Barnea, E., Mairon, R., Edan, Y., and Ben-Shahar, O. (2012). Computer vision for fruit harvesting robots - state of the art and challenges ahead. *International Journal of Computational Vision and Robotics*, 3(1/2):4–34.

Kappler, D., Bohg, J., and Schaal, S. (2015). Leveraging big data for grasp planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311.

Kleeberger, K., Bormann, R., Kraus, W., and Huber, M. F. (2020). A survey on learning-based robotic grasping. *Current Robotics Reports*, 1(4):239–249.

Koirala, A., Walsh, K. B., Wang, Z., and McCarthy, C. (2019). Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. *Precision Agriculture*, 20(6):1107–1135.

Kong, H. and Sukkarieh, S. (2018a). Metamorphic moving horizon estimation. *Automatica*, 97:167–171.

Kong, H. and Sukkarieh, S. (2018b). Suboptimal receding horizon estimation via noise blocking. *Automatica*, 98:66–75.

Kurtser, P. and Edan, Y. (2018a). Statistical models for fruit detectability: Spatial and temporal analyses of sweet peppers. *Biosystems Engineering*, 171:272–289.

Kurtser, P. and Edan, Y. (2018b). The use of dynamic sensing strategies to improve detection for a pepper harvesting robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8286–8293.

Ladicky, L., Shi, J., and Pollefeys, M. (2014). Pulling things out of perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 89–96.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 239–248.

Lehnert, C., English, A., McCool, C., Tow, A., and Perez, T. (2017). Autonomous sweet pepper harvesting for protected cropping systems. *IEEE Robotics and Automation Letters*, 2(2):872–879.

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2018). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436.

Li, J., Karkee, M., Zhang, Q., Xiao, K., and Feng, T. (2016). Characterizing apple picking patterns for robotic harvesting. *Computers and Electronics in Agriculture*, 127:633–640.

Liao, Y., Huang, L., Wang, Y., Kodagoda, S., Yu, Y., and Liu, Y. (2017). Parse geometry from a line: Monocular depth estimation with partial laser observation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5059–5066.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2020). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2015). Microsoft COCO: Common Objects in Context. *arXiv:1405.0312 [cs]*.

Lippiello, V., Siciliano, B., and Villani, L. (2007). Adaptive extended Kalman filtering for visual motion estimation of 3D objects. *Control Engineering Practice*, 15(1):123–134.

Looney, N. E. (1993). Improving fruit size, appearance, and other aspects of fruit crop "quality" with plant bioregulating chemicals. In *International Symposium on Plant Growth Regulators in Fruit Production*. ISHS Acta Horticulturae.

Lundell, J., Verdoja, F., and Kyrki, V. (2019). Robust grasp planning over uncertain shape completions. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1526–1532.

Ma, F. and Karaman, S. (2018). Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8.

Ma, R. and Dollar, A. (2017). Yale OpenHand project: Optimizing open-source hand designs for ease of fabrication and adoption. *IEEE Robotics Automation Magazine*, 24(1):32–40.

Mackay, D. J. C. (1992). A practical bayesian framework for backprop networks. *Neural computation*.

Maldonado, W. and Barbosa, J. C. (2016). Automatic green fruit counting in orange trees using digital images. *Computers and Electronics in Agriculture*, 127:572–581.

Martin, P., Randall, L., and Jackson, T. (2020). Labour use on Australian agriculture. Technical report, Australian Bureau of Agricultural Resource Economics and Sciences (ABARES).

Mason, M. T. and Salisbury, J. K. (1985). *Robot Hands and the Mechanics of Manipulation*. The MIT Press, Cambridge, Mass.

Mehra, R. (1970). On the identification of variances and adaptive Kalman filtering. *IEEE Transactions on Automatic Control*, 15(2):175–184.

Mehta, S. S. and Burks, T. F. (2014). Vision-based control of robotic manipulator for citrus harvesting. *Computers and Electronics in Agriculture*, 102:146–158.

Mehta, S. S., MacKunis, W., and Burks, T. F. (2016). Robust visual servo control in the presence of fruit motion for robotic citrus harvesting. *Computers and Electronics in Agriculture*, 123:362–375.

Mehta, S. S., Ton, C., Asundi, S., and Burks, T. F. (2017). Multiple camera fruit localization using a particle filter. *Computers and Electronics in Agriculture*, 142:139–154.

Mika, A., Buler, Z., Rabcewicz, J., Białkowski, P., and Konopacka, D. (2015). Suitability of plum and prune cultivars, grown in a high density planting system, for mechanical harvesting with a canopy contact, straddle harvester. *Journal of Horticultural Research*, 23(2):69–81.

Morrison, D., Corke, P., and Leitner, J. (2018). Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. In *Robotics: Science and Systems (RSS)*, Pittsburgh, Penn.

Napier, J. R. (1956). The prehensile movements of the human hand. *The Journal of Bone and Joint Surgery. British Volume*, 38-B(4):902–913.

Nguyen, T., Keresztes, J., Vandevoorde, K., Kayacan, E., De Baerdemaeker, J., and Saeys, W. (2014). Apple detection algorithm for robotic harvesting using a RGB-D camera. In *Computer Methods in Applied Mechanics and Engineering*.

Nguyen, T. T., Vandevoorde, K., Wouters, N., Kayacan, E., De Baerdemaeker, J. G., and Saeys, W. (2016). Detection of red and bicoloured apples on tree with an RGB-D camera. *Biosystems Engineering*, 146:33–44.

Odelson, B. J., Rajamani, M. R., and Rawlings, J. B. (2006). A new autocovariance least-squares method for estimating noise covariances. *Automatica*, 42(2):303–308.

OECD and Food and Agriculture Organization of the United Nations (2019). *OECD-FAO Agricultural Outlook 2019-2028*. OECD-FAO Agricultural Outlook. OECD.

Quispe, A. H., Milville, B., Gutiérrez, M. A., Erdogan, C., Stilman, M., Christensen, H., and Amor, H. B. (2015). Exploiting symmetries and extrusions for grasping household objects. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3702–3708.

Rajamani, M. R. and Rawlings, J. B. (2007). Application of a new data-based covariance estimation technique to a nonlinear industrial blending drum. Technical Report 2007-03, University of Texas-Wisconsin.

Rajamani, M. R. and Rawlings, J. B. (2009). Estimation of the disturbance structure from data using semidefinite programming and optimal weighting. *Automatica*, 45(1):142–148.

Ramon Soria, P., Sukkar, F., Martens, W., Arrue, B. C., and Fitch, R. (2018). Multi-view probabilistic segmentation of pome fruit with a low-cost rgb-d camera. In Ollero, A., Sanfeliu, A., Montano, L., Lau, N., and Cardeira, C., editors, *ROBOT 2017: Third Iberian Robotics Conference*, Advances in Intelligent Systems and Computing, pages 320–331, Cham. Springer International Publishing.

Rao, C. V., Rawlings, J. B., and Lee, J. H. (2001). Constrained linear state estimation—a moving horizon approach. *Automatica*, 37(10):1619–1628.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149.

Rennie, C., Shome, R., Bekris, K., and De Souza, A. (2015). A dataset for improved RGBD-based object detection and pose estimation for warehouse pick-and-place. *IEEE Robotics and Automation Letters*, 1.

Rimon, E. and Burdick, J. (1996). On force and form closure for multiple finger grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1795–1800 vol.2.

Ringdahl, O., Kurtser, P., and Edan, Y. (2019). Evaluation of approach strategies for harvesting robots: Case study of sweet pepper harvesting. *Journal of Intelligent & Robotic Systems*, 95(1):149–164.

Roa, M. A. and Suárez, R. (2015). Grasp quality measures: Review and performance. *Autonomous Robots*, 38(1):65–88.

Rock, J., Gupta, T., Thorsen, J., Gwak, J., Shin, D., and Hoiem, D. (2015). Completing 3D object shape from one depth image. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2484–2493.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., and McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, 16(8):1222.

Sa, I., Lehnert, C., English, A., McCool, C., Dayoub, F., Upcroft, B., and Perez, T. (2017). Peduncle detection of sweet pepper for autonomous crop harvesting—Combined color and 3-D information. *IEEE Robotics and Automation Letters*, 2(2):765–772.

Schlotfeldt, B., Atanasov, N., and Pappas, G. J. (2019). Maximum information bounds for planning active sensing trajectories. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4913–4920.

Schlotfeldt, B., Thakur, D., Atanasov, N., Kumar, V., and Pappas, G. J. (2018). Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032.

Sengupta, S. and Lee, W. S. (2014). Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions. *Biosystems Engineering*, 117:51–61.

Silwal, A., Davidson, J. R., Karkee, M., Mo, C., Zhang, Q., and Lewis, K. (2017). Design, integration, and field evaluation of a robotic apple harvester. *Journal of Field Robotics*, 34(6):1140–1159.

Stein, M., Bargoti, S., and Underwood, J. (2016). Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors*, 16(11):1915.

Sun, Y., Song, Y. S., and Paik, J. (2013). Characterization of silicone rubber based soft pneumatic actuators. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4446–4453.

Sun, Y., Yap, H. K., Liang, X., Guo, J., Qi, P., Ang, M. H., and Yeow, C.-H. (2017). Stiffness customization and patterning for property modulation of silicone-based soft pneumatic actuators. *Soft Robotics*, 4(3):251–260.

Suwajanakorn, S., Hernandez, C., and Seitz, S. M. (2015). Depth from focus with your mobile phone. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3497–3506.

Tanigaki, K., Fujiura, T., Akase, A., and Imagawa, J. (2008). Cherry-harvesting robot. *Computers and Electronics in Agriculture*, 63:65–72.

Tu, S., Pang, J., Liu, H., Zhuang, N., Chen, Y., Zheng, C., Wan, H., and Xue, Y. (2020). Passion fruit detection and counting based on multiple scale faster R-CNN using RGB-D images. *Precision Agriculture*.

van Henten, E., Hemming, J., van Tuijl, B., Kornet, J., Meuleman, J., Bontsema, J., and van Os, E. (2002). An autonomous robot for harvesting cucumbers in greenhouses. *Autonomous Robots*, 13(3):241–258.

van Henten, E. J., Bac, C. W., Hemming, J., and Edan, Y. (2013). Robotics in protected cultivation. *IFAC Proceedings Volumes*, 46(18):170–177.

Varley, J., DeChant, C., Richardson, A., Ruales, J., and Allen, P. (2017). Shape completion enabled robotic grasping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2442–2447.

Verl, A., Albu-Schäffer, A., Brock, O., and Raatz, A., editors (2015). *Soft Robotics: Transferring Theory to Application*. Springer-Verlag, Berlin Heidelberg.

Vezzani, G., Pattacini, U., and Natale, L. (2017). A grasping approach based on superquadric models. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1579–1586.

Vitzrabin, E. and Edan, Y. (2016). Adaptive thresholding with fusion using a RGBD sensor for red sweet-pepper detection. *Biosystems Engineering*, 146:45–56.

Vougioukas, S. G., Arikapudi, R., and Munic, J. (2016). A study of fruit reachability in orchard trees by linear-only motion. *IFAC-PapersOnLine*, 49(16):277–280.

Wallace, N., Kong, H., Hill, A., and Sukkarieh, S. (2019a). Experimental validation of structured receding horizon estimation and control for mobile ground robot slip compensation. *Field and Service Robotics*, pages 1–16.

Wallace, N. D., Kong, H., Hill, A. J., and Sukkarieh, S. (2019b). Receding horizon estimation and control with structured noise blocking for mobile robot slip compensation. *2019 International Conference on Robotics and Automation (ICRA)*.

Wang, Q., Nuske, S., Bergerman, M., and Singh, S. (2013). Automated crop yield estimation for apple orchards. In Desai, J. P., Dudek, G., Khatib, O., and Kumar, V., editors, *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, Springer Tracts in Advanced Robotics, pages 745–758. Springer International Publishing, Heidelberg.

Williams, C. (1996). Computing with infinite networks. In *Advances in Neural Information Processing Systems 9*, pages 295–301. MIT Press.

Wilson, W. J., Hulls, C. C. W., and Bell, G. S. (1996). Relative end-effector control using cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5):684–696.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3D ShapeNets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920.

Xiong, Y., Ge, Y., and From, P. J. (2020). An obstacle separation method for robotic picking of fruits in clusters. *Computers and Electronics in Agriculture*, 175:105397.

Xiong, Y., Peng, C., Grimstad, L., From, P. J., and Isler, V. (2019). Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper. *Computers and Electronics in Agriculture*, 157:392–402.

Xu, D., Ricci, E., Ouyang, W., Wang, X., and Sebe, N. (2018). Monocular depth estimation using multi-scale continuous CRFs as sequential deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Zhao, C., Sun, Q., Zhang, C., Tang, Y., and Qian, F. (2020). Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences*, 63(9):1612–1627.

Zhao, Y., Gong, L., Huang, Y., and Liu, C. (2016). A review of key techniques of vision-based control for harvesting robot. *Computers and Electronics in Agriculture*, 127:311–323.