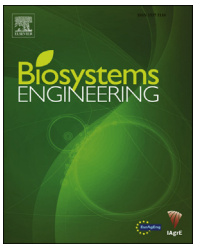
BIOSYSTEMS ENGINEERING XXX (2016) 1–14

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/issn/15375110



Special Issue: Robotic Agriculture

Research Paper

Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation

Ruud Barth [a,b,*], *Jochen Hemming* [b], *Eldert J. van Henten* [c]

[a] *Harvard University, 60 Oxford Street, Cambridge, Massachussets, United States*
[b] *Wageningen University & Research Centre, Greenhouse Horticulture, P.O. Box 644, 6700 AP, Wageningen, The Netherlands*
[c] *Wageningen University & Research Centre, Farm Technology Group, Droevendaalsesteeg 1, 6708 PB, Wageningen, The Netherlands*

ARTICLE INFO

Article history:
Published online xxx

Keywords:
Framework
Harvest robots
Visual servo control
ROS
SLAM

A modular software framework design that allows flexible implementation of eye-in-hand sensing and motion control for agricultural robotics in dense vegetation is reported. Harvesting robots in cultivars with dense vegetation require multiple viewpoints and on-line trajectory adjustments in order to reduce the amount of false negatives and correct for fruit movement. In contrast to specialised software, the framework proposed aims to support a wide variety of agricultural use cases, hardware and extensions. A set of Robotic Operating System (ROS) nodes was created to ensure modularity and separation of concerns, implementing functionalities for application control, robot motion control, image acquisition, fruit detection, visual servo control and simultaneous localisation and mapping (SLAM) for monocular relative depth estimation and scene reconstruction. Coordination functionality was implemented by the application control node with a finite state machine. In order to provide visual servo control and simultaneous localisation and mapping functionalities, off-the-shelf libraries Visual Servoing Platform library (ViSP) and Large Scale Direct SLAM (LSD-SLAM) were wrapped in ROS nodes. The capabilities of the framework are demonstrated by an example implementation for use with a sweet-pepper crop, combined with hardware consisting of a Baxter robot and a colour camera placed on its end-effector. Qualitative tests were performed under laboratory conditions using an artificial dense vegetation sweet-pepper crop. Results indicated the framework can be implemented for sensing and robot motion control in sweet-pepper using visual information from the end-effector. Future research to apply the framework to other use-cases and validate the performance of its components in servo applications under real greenhouse conditions is suggested.

* *Corresponding author*. Wageningen University & Research Centre, Greenhouse Horticulture, P.O. Box 644, 6700 AP, Wageningen, The Netherlands.
E-mail addresses: r.barth@wur.nl (R. Barth), jochen.hemming@wur.nl (J. Hemming), eldert.vanhenten@wur.nl (E.J. van Henten).

http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

Nomenclature

s^*	desired image feature vector, –
s	current image feature vector, –
L_s	interaction matrix that links variation of features over time and camera velocity, –
$\widehat{L_s}$	approximation of interaction matrix, –
$\widehat{L_s^+}$	pseudo-inverse of approximation of interaction matrix, –
λ	proportional coefficient of the exponential convergence of the error, –
v	camera velocity or kinematics screw vector
R	red values of image, –
G	green values of image, –
B	blue values of image, –
X	projection of RGB colourspace to novel X-axis, –
Y	projection of RGB colourspace to novel Y-axis, –
Z	projection of RGB colourspace to novel Z-axis, –
l	Lightness dimension of CIELab colourspace, –
a	colour-opponent dimension a of CIELab colourspace, negative values indicate green while positive values indicate magenta, –
b	colour-opponent dimension b of CIELab colourspace, negative values indicate blue and positive values indicate yellow, –
x	horizontal position of feature in camera frame, m
y	vertical position of feature in camera frame, m
z	depth position of feature in camera frame, m
u_0	x-coordinate of image centre, pixels
v_0	y-coordinate of image centre, pixels
u_i	horizontal position pixel in image, pixels
v_i	vertical position pixel in image, pixels
p_x	horizontal pixel size on camera sensor, m
p_y	vertical pixel size on camera sensor, m

1. Introduction

During the design of robotic applications for harvesting horticultural products, two key challenges need to be solved. The first is the detection of a target location of the fruit. The second is moving the end-effector towards that location with precision to perform a harvest action. There are several ways to address each of these challenges. For example, one approach solves fruit detection by using one or few viewpoints from a sensing module located externally from the robot. However, in crops with a high vegetation density, using a low number of viewpoints results in false negatives due to a large amount of occluding leaves and branches (Hemming, Ruizendaal, Hofstee, & Van Henten, 2014b). Furthermore, the external placement of the sensor(s) requires one or multiple frame transformations. Slight errors therein accumulate, resulting in inaccurate target coordinates. When a location of the target fruit is acquired, moving the end-effector there can be solved by executing a planned motion trajectory without additional sensing. However, dislocation of the target can occur as the robot enters and interacts with a dense crop. Both the frame transformation errors and dislocation of the target can result in poor end-effector placement at the target (Hemming et al., 2014a; Henten et al., 2003). An example implementation of external sensing and planned motion control was tested during the European 7th Framework Programme project Clever Robots for Crops (CROPS) (GA no. 246252). During this project, a proof-of-principle harvesting robot was created for a dense sweet-pepper crop. A sensing module dislocated from the robot provided fruit detection from a single viewpoint. Thereafter a motion trajectory was executed without further sensing. It was concluded that this approach was one of the causes of low harvest performance, both in cycle time as well as in fruit detection rates (Bac, 2015). Another example of a cucumber harvesting robot uses a similar approach (Van Henten et al., 2002), where a single viewpoint in the workspace of the robot provided fruit positions. In both the CROPS and cucumber robot, additional sensing could be performed to refine fruit positions with a second set of cameras on the end-effector. However, in both field tests this feature was not used. This extra single sensing step before the final motion execution is also known as look-and-move (Hutchinson, Hager, & Corke, 1996). When a camera is attached to the end-effector, it is often named an eye-in-hand sensor (Hutchinson et al., 1996). For a strawberry harvesting robot, a similar eye-in-hand look-and-move approach was used (Hayashi et al., 2010).

A different approach is to solve fruit detection and motion control using primarily eye-in-hand sensing. External sensors are not necessarily excluded in this paradigm, though the application is not dependent on this additional secondary sensing source. For the fruit detection, the internal location of the sensor(s) reduces the number of coordinate frame transformations to a single one. Moreover it allows the application to sense the scene from multiple viewpoints with pose changes of the end-effector, expected to decrease the number of false negative detections in a dense crop. For the motion towards the target, this approach allows for continuous incremental visual feedback and corrections, also known as visual servo control (Hutchinson et al., 1996; Marchand, Spindler, & Chaumette, 2005). Examples of robotic harvesters in horticulture using visual servo control are numerous. For a sweet-pepper harvesting robot in Japan, a visual servo control algorithm positioned the end-effector near the fruit using stereo images (Kitamura & Oka, 2005). Although the camera was not part of the end-effector, it was placed within its workspace and aligned with the optical axis. In a strawberry harvesting robot, a set of external sensors first provided rough fruit position after which an eye-in-hand system moved towards it using visual servo control (Han et al., 2012). For applications of an apple harvesting robot (De-An, Jidong, Wei, Ying, & Yu, 2011) and a citrus harvesting robot (Mehta & Burks, 2014), eye-in-hand visual servo control systems were created. Another application for an apple harvesting robot also applied eye-in-hand sensing, however did not implement a full visual servo control. Instead look-and-move corrections were performed multiple times during the fruit approach (Baeten, Donn, Boedrij, Beckers, & Claesen, 2008).

The aim of our research was to provide a flexible modular framework for eye-in-hand sensing and motion control in robotic harvest applications as a standardised approach. In the aforementioned previous research, the designs of sensing

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

and visual servo control algorithms were ad hoc and application specific, therefore hard to migrate to other use-cases. Our approach aims to provide a consistent approach, designed to cope with a wide variety of applications. The framework is primarily designed for dense crops in agri- and horticultural robotics, where a single viewpoint is not sufficient for sensing. Other frameworks for sensing and visual servo control focus on the design of a single low level function (Bachiller, Cerrada, & Cerrada, 2003; Jara, Pomares, Candelas, & Torres, 2014; Mahony, 2011; Marchand et al., 2005; Wu, Lou, Chen, Hirche, & Kuhnlenz, 2010). Our aim is to provide a higher level framework architecture that spans the functionality required for a full robot application, as suggested in previous research (Bachiller, Cerrada, & Cerrada, 2006).

This paper firstly provides the general design of the framework in Section 2.1 by describing the required functionalities and architecture of the software and its components. An example implementation for a sweet-pepper use case is then described in Section 3, along with qualitative tests under laboratory conditions. The primary aim of these tests was to i) demonstrate that the framework can execute an eye-in-hand sensing and visual servo control sequence and to ii) extend the functionality of sensing with 3D scene reconstruction. The performance of eye-in-hand sensing and motion control libraries are not validated. Section 4 describes the results of our research followed by a discussion in Section 5.

2. Materials

2.1. *Software*

The functions of a robot can be divided into three broad primitives: sensing, planning and acting (Murphy, 2000). To organise the robotic behaviour with these primitives, one of several paradigms can be implemented in a software architecture. For a visual servo control task, a reactive paradigm is most applicable because it routes sensor information directly to actions. However, this omits any planning that an application may need. The hybrid deliberative/reactive paradigm introduces the planning primitive whilst also supporting reactive behaviour. In this paradigm, a global planner executes sub-tasks that can be either planned or reactive, acting as an intermediate coordinator of sensing information (Murphy, 2000). For our framework this paradigm was chosen because both planning and reactive tasks were used.

A software architecture, or framework, that implements the hybrid deliberative/reactive paradigm should describe a set of components and their interaction (Dean & Wellman, 1991). For our framework five required functionalities were differentiated that fall into the three primitives of sensing, planning and/or acting: (i) image acquisition, (ii) fruit detection, (iii) application control, (iv) visual servo control and (v) robot control. The functions (i), (iii) and (v) fall into a single primitive. However, functions (ii) and (iv) overlap in the planning primitive because they also process, analyse and plan with data.

In Fig. 1 an overview of the functions for the framework is provided, divided over the robotic primitives.

Flexibility of the framework results from the functional implementation in independent modules. Through such a design pattern, functionality is replaceable and expandable with new features without revisions of other modules. This in contrast to creating a single library that entangles all functionality, resulting in poor affordance to substitution of components and a limited separation of concerns (Felix & Ortin, 2014).

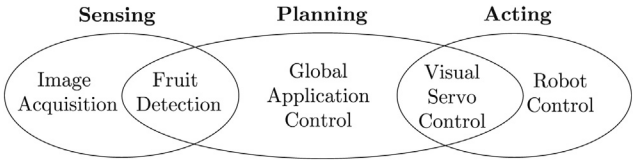


Fig. 1 – Venn diagram of the framework's required functions, divided over the robot behaviour primitives sensing, planning and acting.

The functions were implemented in the middleware 'Robotic Operating System' (ROS) (Quigley et al., 2009). ROS allows the creation of a modular networks of nodes that perform dedicated subsets of the computation and organises the communication between them. Furthermore, a shared stack of robotic libraries are available to all nodes to facilitate computations for robotics, such as for timing, coordinate frame transformations and robot motion simulations. ROS also provides a set of basic communication policies such as services, publishers and subscriptions as well as a more advanced policy where actions can be monitored or pre-empted during a continuous feedback loop.

In Fig. 2 the suggested interaction architecture between the ROS nodes of the framework is displayed. Functionalities in the framework were explicitly separated to facilitate replacement of nodes and the extension of new functionalities. Furthermore, centralised functionalities avoid functional duplication across nodes. An additional function of simultaneous localisation and mapping (SLAM) was added to show the extensibility of the framework. The central position

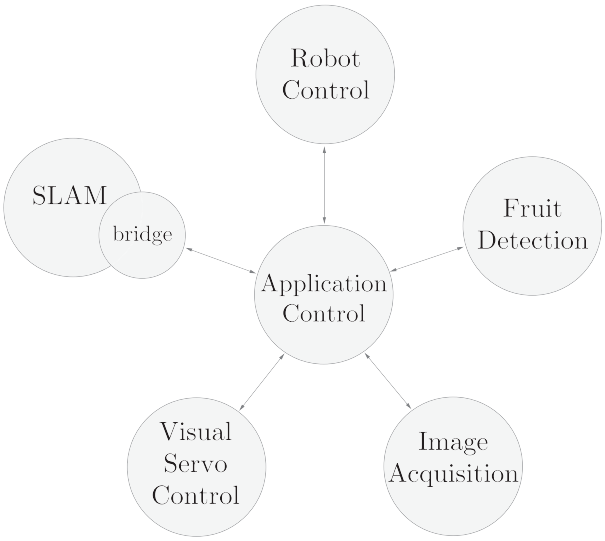


Fig. 2 – ROS nodes architecture of the eye-in-hand sensing and motion control framework. Links indicate communication interactions.

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

of the application control node in the communication allows for flexible coordination as opposed to distributed control over several interacting nodes. In the following sections, the required functionality of each node will described in detail.

2.1.1. Application control

The application control node fullfilled a coordinating role by communicating with all other nodes and processing their feedback information. This goal was achieved in this node by implementing a finite state machine (FSM) based on previous research (Barth et al., 2014; Hellstrom & Ringdahl, 2013), which has similarity with other FSM approaches like ROS Commander (Nguyen, Ciocarlie, Hsiao, & Kemp, 2013) and SMACH (Bohren & Cousins, 2010). The FSM is a modular collection of states and their respective state transitions. In each state a certain subtask of the robot application can be executed. The use of a state machine gave the framework another layer of flexibility by facilitating the reuse of states, smooth addition of states and rerouting of transitions without requiring recompilation of the framework. The concern of coordination was separated in this node from the other nodes.

2.1.2. Image acquisition

The image acquisition node provided the framework with the functionality of creating a connection to a camera and grabbing colour and monochrome images upon request. ROS required the images to be sent in the ROS image format. The node requires exposure settings and gains for each channel, which can be set in the ROS launch file. To obtain the gain parameters, a colour calibration procedure should be performed by manually adjusting the gain levels until the red, green and blue values of all pixels are equal given a recorded image of a grey calibration reference object. Rectification of the images before sending is required to allow a relation between image coordinates in pixels and real world coordinates in metres accordingly. For this, the camera parameters should be known.

2.1.3. Fruit detection

The goal of the fruit detection node was to provide information about the fruit in a given image. Note that the function of fruit detection is a broad term, to which at least 3 subfunctions can be distinguished that are relevant for harvesting robots: finding fruit, localising fruit in 3D and determining ripeness and/or harvestability. Depending on which sub-functions are required by the application, each subfunction should provide a service that returns a set of features of an image. These features can be descriptive, like surface areas, or geometrical like the position of the largest fruit in the image. In this framework, the visual servo control constrains the image analysis computation time and should be below 100 ms.

2.1.4. Visual servo control

The functionality provided by this node was to use image features to control the motion of a robot, using a continuous correctional feedback loop (Hutchinson et al., 1996) or online trajectory generation (Kröger, 2010). Image information could be used from one of more cameras, either located on the gripper or external from the robot. Independent of the camera configuration, the task required a set of geometrical visual features s to be extracted from the acquired image(s). In our framework the fruit detection node provided this functionality. To use the geometrical features for correcting the motion of the robot, a control law must be designed that realises the desired feature values s^* by minimising the error $(s - s^*)$. For this an interaction matrix L_s, also known as the image Jacobian, needs to be approximated that models the relationship between the time variation of the features and the camera velocity v (Marchand et al., 2005). Vector v is also known as the kinematics screw vector, encoding the required variation in pose of the camera relative to the object. The general case of an eye-in-hand control law where camera velocities are computed is defined by:

$$v = -\lambda \widehat{L_s^+}(s - s^*), \tag{1}$$

where λ is the proportional coefficient of the exponential convergence of the error and $\widehat{L_s^+}$ is the pseudo-inverse of the estimation of the interaction matrix, which is parameterised by intrinsic camera parameters (focal length, image sensor format and principal image point) and feature location information (m) relative to the camera frame. To add robot motion control to the framework, the Visual Servoing Platform library (ViSP) was wrapped in a ROS node (Marchand, 1999; Marchand et al., 2005), allowing for rapid prototyping of visual servo control algorithms and specifically developed for high-level applications. With Visp a set of elementary tasks can be created by combining visual features. It is designed to be modular, hardware-independent, extendable and portable, making this library highly suitable as a key component for our highly flexible and modular framework. Under the assumption that visual features are defined upon geometrical primitives, such as points or lines, ViSP can approximate the interaction matrix analytically using a previously proposed method (Espiau, Chaumette, & Rives, 1992).

In each iteration of the servo control loop, this ROS node computes the velocity vector v given (i) a set of desired geometric features s^*, (ii) a set of current geometric features s and (iii) the convergence coefficient, ranging from 0 to 1. The velocity vector encodes the required change in pose applied to the end-effector to converge to the desired feature values. Note that in some cases convergence and stability problems may occur (Chaumette, 1998).

2.1.5. Robot control

The robot control node provided the functionality to move the end-point of a robot to a desired pose, receiving a real-world Cartesian coordinate and returning a movement status. Visual servo control requires updating the end-effector goal pose multiple times per second. This can be achieved by goals that can be pre-empted or when joint motions of the robot are directly accessible. In Section 3 the way this was implemented in the Baxter robot is described.

2.1.6. Simultaneous localisation and mapping

The aim of this node is to provide additional sensing information from images by implementing a simultaneous localisation and mapping (SLAM) method. Largely used for

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

unknown and unmapped environments, this approach allows for camera pose estimations and three-dimensional scene reconstruction (Durrant-Whyte & Bailey, 2006). From this information, relative depth between objects in the scene can be derived.

For this purpose, the Large Scale Direct SLAM (LSD-SLAM) was used (Engel, Schps, & Cremers, 2014). In contrast to other methods, it runs real-time on modern CPU's and uses a featureless approach working directly on monocular image intensities. The library was already wrapped in a ROS Indigo node, hence no modifications for our framework were required. However, by default only the visual odometry (estimated camera pose) in combination with relative depth map keyframes were published in the ROS system. Therefore the three-dimensional reconstruction was not available outside the LSD-SLAM node. To add this feature in ROS, a bridge node was implemented which concatenated multiple depth keyframes after transformation to the same world frame using the SLAM's published odometry information.

2.2. *Hardware*

The hardware used for testing the framework consisted of a camera attached to the end-effector of a robot. The camera was connected to the computer through USB and the robot was connected to the computer through an ethernet connection.

2.2.1. *Robot*

The framework was applied and tested on a Baxter robot by Rethink Robotics (Fitzgerald, 2013), depicted in Fig. 3. The robot was designed to mimic and replace workers on a production line, performing tasks such as sorting or picking and placing parts. The human sized robot has 2 mirrored 7 degrees of freedom arms, although only one was used in our setup. The robot was chosen for its native ROS support. Baxter runs a ROS master core to which target joint angles can be published, executed by an internal controller. Baxter also provides inverse kinematics (IK) service for calculating joint angles given a 3 dimensional Cartesian coordinate relative to the robot frame. Furthermore, Baxter publishes the pose of the end-effector and all joints. The standard Baxter end-effector was used.

Fig. 3 – Baxter robot by Rethink Robotics, with 2 mirrored 7 degree of freedom arms.

2.2.2. *Camera*

A USB CMOS colour Autofocus Camera (DFK 72AUC02-F, TheImagingSource, Germany) was attached on top of the tool centre point of the robot, to which the standard end-effector was also mounted. Images were grabbed by the ROS image acquisition node with a rolling shutter at a resolution of 640 × 480 pixels. A M12x0.5 mount lens with a focal length of 4.6 mm was attached. Exposure was set to 50 ms, allowing for a frame rate of 20 images s^{-1}. The autofocus feature was not available under the Linux operating system and was not used.

2.2.3. *Computer*

The framework was run on a MacBook Pro, 2.4 GHz Intel Core i5 with 8 GB of DDR3 memory operating on Ubuntu 12.04 Precise Pangolin.

3. Methods

To validate the design of the eye-in-hand sensing and motion control framework, we implemented the software functionalities described in Section 2.1 with the hardware described in Section 2.2 for the dense sweet-pepper crop use case. For this purpose, nodes for the robot control, image acquisition, fruit detection, visual servo control and the application control were implemented to provide the required functionality. All nodes were implemented in C++ ROS, version Indigo. For the image acquisition and fruit detection nodes, the industrial machine vision library MVtec Halcon 11.0 (MVTec Software GmbH, 2015) was used by a wrapped ROS Indigo node around the Halcon HDevEngine. Upon initialisation of the ROS nodes, a set of custom Halcon procedures were loaded into memory. The functions were not hardcoded in the source, but specified in the ROS launch file, allowing functions to be updated or replaced without recompilation of the framework. Note that open source image processing libraries, e.g. OpenCV (Culjak, Abram, Pribanic, Dzapo, & Cifrek, 2012), can replace the commercially licensed Halcon library with minor effort. The framework was tested with the hardware in combination with an artificial dense sweet-pepper crop under laboratory conditions. In the following section, the use case is further specified, first describing the use-case specific software implementation of the framework, followed by the experimental setup of the laboratory tests.

3.1. *Use case description*

Sweet-pepper (*Capsicum annum*) is a high value crop, which is currently manually harvested in high wired greenhouse cultivation systems. Due to their organised, repetitive structure, as seen on the left in Fig. 4 these systems are suitable for adding robots. Unlike other crops, the fruit visibility is low in a single viewpoint due to occlusions by other plant parts (Hemming et al., 2014b). This can be seen in the right image of Fig. 4, where on the foreground a ripe sweet pepper is occluded by the stem and wire and a green pepper is partially occluded by leaves. Furthermore, the location of a sweet-pepper can be ambiguous, as a red patch in an image can either be a wholly visible sweet-pepper in the background or a highly occluded sweet-pepper in the foreground.

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

Fig. 4 – Example photographs of a sweet-pepper crop in a Dutch high-wire greenhouse cultivation system. In the left image a front view is shown. Double plant rows are separated by a workspace with a rail system to allow access to the plants. The right image shows a side view taken from the workspace facing towards the plants.

Solving the visibility problem in dense crops requires an approach that uses multiple viewpoints, either before or during the approach to the target fruit. Moreover, whilst executing the motion towards the target, corrections to the path are required because the dense vegetation is easily moved and the target displaced by a robot entering the crop. Target reachability is another issue, as individual plants are spaced between 0.1 and 0.3 m (Bac, 2015). Obstacle avoidance may therefore be required.

3.2. *Experimental setup*

An artificial sweet-pepper crop section was created using plastic imitation fruit and leaves. Although colour and reflective properties were similar to real fruit for the human eye, they differ in other material properties such as hyperspectral information and firmness. However, the materials sufficed for our purposes since only RGB analysis was required. The main nerves of the leaves were fitted with a metal wire, allowing the leaves to be shaped. The leaves and the fruit were attached to a vertically placed thin pole of wood that represented a stem. By shaping the leaves, different degrees of occlusion could be realised. In Fig. 5 a 360° view at 45° increments of a typical setup is displayed. The visibility of the fruit depended on the perspective; the fruit were fully visible in one view and entirely occluded in another. In many views, leaves or stems partially occluded the fruit.

The objective of this experiment was to show that the robot could find and access the fruit. Therefore, it was sufficient that the end-effector stopped just in front of the fruit. This was assured by placing the target fruit just out of reach of the maximum robot arm stretch at 1.05 m. The test crop was placed around 10 various locations, within an arc of around 0.5 m in front of the robot. The workspace towards the front of the robot, and therefore the number of test locations, was limited because the robot's workspace was primarily designed for pick and place operations in the horizontal plane. At each location, the occlusion of the test crop was varied and multiple state machine cycles were executed. Qualitative results of the framework's performance on a dense crop were registered and these results will be discussed in the next section.

3.3. *Use case specific framework implementation*

3.3.1. *Robot control*

Two methods for motion control of the Baxter robot were implemented. The first is a ROS actionlib service, which enabled pre-emptable tasks. With this method the status of longer movement actions could be tracked and aborted, suitable for moving to waypoints. The second method is a direct robot joint angle control, allowing to continuously change the rotation of each individual joint. This method provides short motions that can be updated during the movement, suitable for visual servo control. Both methods call Baxter's inverse kinematics ROS service. For this service, a desired pose in real world coordinates can be specified for the end-effector. The service will return a set of joint angles to move the arm to the target location, or gives feedback when it is unreachable or collisions are expected.

3.3.2. *Image acquisition*

The image acquisition node implemented a connection with the camera through Halcon. A service was provided to send colour or monochrome ROS images upon request. Grabbed Halcon format images were efficiently bridged to the ROS image format before sending. Furthermore this node also visualized grabbed images. In order to rectify the image, Halcon's default procedures were used for multi-view 3D calibration. For this purpose, a set of 100 images was taken of a calibration plate in various locations and orientations. Internal camera parameters were calculated and applied to each new image to remove lens distortion.

3.3.3. *Fruit detection*

During the research project CROPS, an end-effector was developed that did not require the orientation of the fruit nor an exact position thereof for a successful harvest (Van Tuijl, Wais, & Yael, 2013) (Hemming, Van Tuijl, Gauchel, & Wais, 2014c). This reduces constraints on the fruit detection, allowing for more simplistic and fast approaches which are suited for visual servo control. Other approaches that calculate exact poses or use three-dimensional object matching are generally more time consuming and therefore less appropriate for visual servo control. However, such approaches can be effectively applied, for example in grasp synthesis using

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

Fig. 5 – Typical views at 45° increments around an artificial sweet-pepper crop used in the experiment. The backdrop is removed for clarity.

active vision (Çalli, 2015; Çalli, Wisse, & Jonker, 2011; Yazicioglu, Çalli, & Unel, 2009).

A previous approach (Song et al., 2014) for sweet-pepper detection classified image features. A colour based classification provided the regions of interest in multiple images from which maximally stable colour region features (Forssen, 2007) were extracted. Because the computational complexity and temporal performance was not reported, it is unknown if this approach can meet the time constraint in visual servo control.

To implement a simplistic and fast fruit detection for sweet-pepper, an advanced blob detection was created. It started the analysis by converting the image from a RGB to a CIELab colourspace using the equations (2)–(6). Contrary to colourspaces that encode a single axis for colour, CIELab has two axis for colour a,b and one for luminosity l. Because the a axis encodes a spectrum separating green from violet-red, this channel provided distinctive contrast between red sweet-pepper and the green surroundings. Note that for other use cases or colours of sweet-pepper, a transformation to the HSI colourspace might be more suitable.

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.412453 & 0.357580 & 0.180432 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{2}$$

$$l = 116f(Y) - 16 \tag{3}$$

$$a = 500(f(X) - f(Y)) \tag{4}$$

$$b = 200(f(Y) - f(Z)) \tag{5}$$

Where

$$f(t) = \begin{cases} t^{1/3} & \text{if } t > \left(\frac{6}{29}\right)^3 \\ \frac{1}{3}\left(\frac{29}{6}\right)^2 t + \frac{4}{29} & \text{otherwise} \end{cases} \tag{6}$$

For segmentation of the sweet-pepper blobs, the a channel values ranging from 25 to 70 were selected. This segmentation generally contains noise, which was filtered out by a region opening operation (equal to dilation of an erosion) using a 5 pixel round element, twice as large as the noise to be filtered out. From the largest remaining region, the size and image coordinates of the centre of gravity were calculated and returned to the application control node as features. In Fig. 6 intermediate results of the image processing pipeline applied to an example image are displayed.

3.3.4. *Visual servo control*

To implement visual servo control for this use case, geometric features needed to be defined. However, high occlusion rates restrict the approach of deriving an object pose as a reliable feature. Instead, only parts of the fruit can be seen from a subset of all viewpoints. The centre of gravity image coordinates of the largest segmented sweet-pepper part was returned as a feature, as described in Section 3.3.3. This can be used as a geometrical feature as it defines a point in two dimensional space. The desired value of this feature was set in the centre coordinates of the image.

In this application the control law for image-based visual servo control and eye-in-hand tasks in Eq. (1) is used. In ViSP the estimation of the interaction matrix for a 2D image feature is given by:

$$\widehat{L}_s = \begin{bmatrix} -1/z & 0 & x/z & yx & -(1+x^2) \\ 0 & -1/z & y/z & 1+y^2 & -xy \end{bmatrix}, \tag{7}$$

where z is either a known or estimated feature depth in the camera frame. In our application, the estimation of z had a starting value of 0.40 m, as this was the starting distance of the crop scanning as described in Section 3.3.5. This value should be updated in each visual servo cycle.

The interaction matrix also requires the positions of visual features expressed in metres rather than image pixel coordinates. For this the previously obtained camera parameters (Section 2.1.2) were used for a perspective projection without distortion model. The parameters were x-coordinate of image centre u_0, y-coordinate of image centre v_0, horizontal sensor pixel size p_x and vertical sensor pixel

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

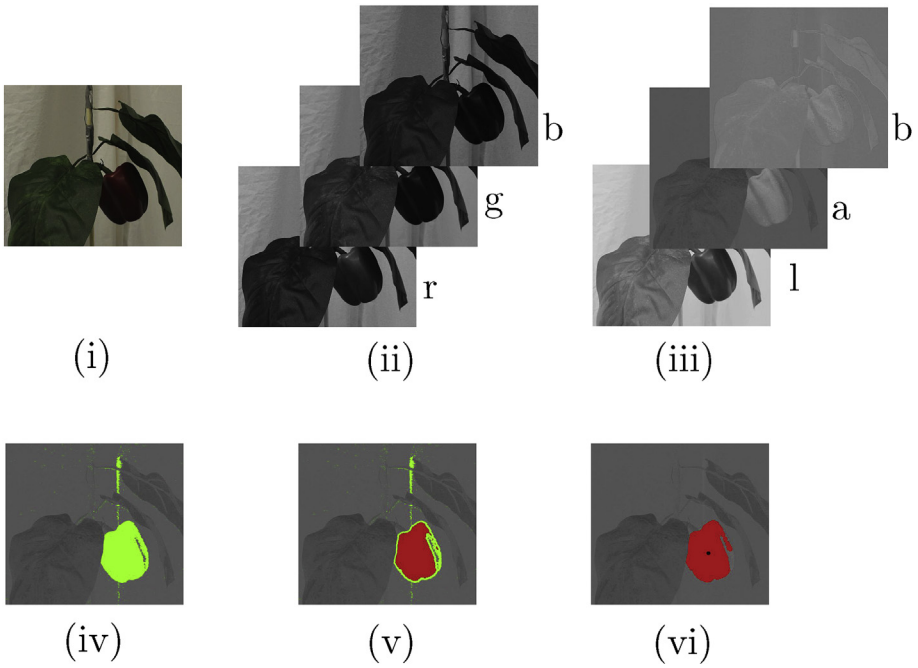


Fig. 6 – Intermediate results of the sweet-pepper detection pipeline on an example colour image (i). (ii) The image is separated in channels for red (r), green (g) and blue (b). (iii) RGB channels are converted to CIELab channels *l*, *a* and *b*. (iv) A threshold operation on channel *a* segments the sweet-pepper and some noise. (v) Noise removal by region erosion operation using an element twice as large as the noise. (vi) A region dilation operation of the same size as the shrinking operation segments the whole pepper. The size and centre of this region are returned as result features. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

size p_y. If we define (u_i,v_i) as the position of a pixel in the image, then the position of that pixel in metres in the camera frame can be obtained by:

$$x = (u_i - u_0)/p_x \tag{8}$$

$$y = (v_i - v_0)/p_y \tag{9}$$

The coefficient of the exponential convergence of the error λ in the control law was set to the default value of 0.3.

3.3.5. *Application control*

The application control node's FSM was implemented for the sweet-pepper use case. Six states were created that each executed a sub-task in the program. The states and their transitions are displayed in Fig. 7.

The program started in the ColdBoot state where all hard- and software modules were initialised. When all modules were initialised, the state machine advanced to the Ready state that waited for an external trigger to start a harvest cycle. First, the robot moved to an initial home position,

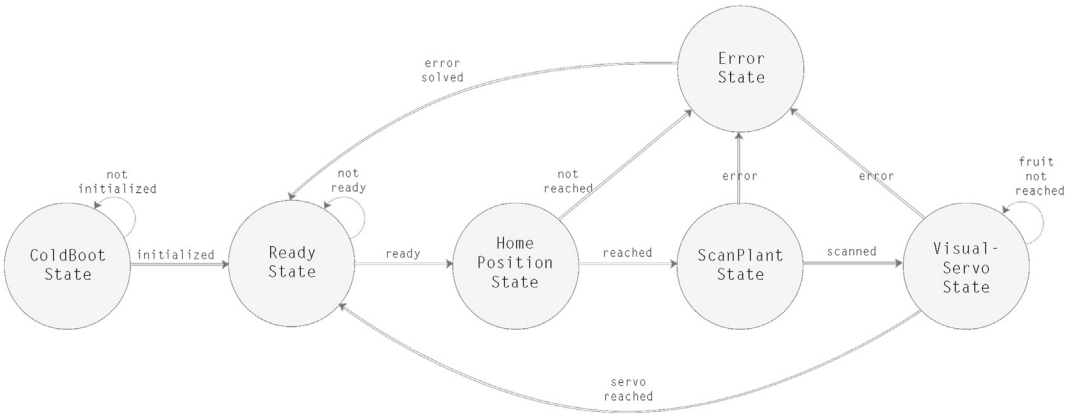


Fig. 7 – Flowchart of states and their transitions that were implemented in the finite state machine of the application control node.

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

defined as 0.40 m in front of the plant. When this position was reached, the state machine started a sensing procedure in the ScanPlant state. During this procedure the end-effector moved to predefined waypoints in the horizontal plane whilst the end-effector remained facing the plant. In Fig. 8 a visual representation of the procedure is shown. The waypoints were chosen following from a set of constraints consisting of i) the robot workspace, ii) representative greenhouse conditions such as a maximum distance of 0.40 m from the target and a restricted maximum angle of approach around 90° and iii) to have at least one viewpoint with a fully occluded fruit and one viewpoint with a fully visible fruit. The motion planning involved during the plant scanning phase is a closed-loop execution of a planned motion trajectory, provided by the inverse kinematics solver of Baxter as described in Section 2.2.1.

In parallel, the image acquisition node was continuously triggered at 20 Hz to obtain images for (i) the SLAM node and (ii) the fruit detection node. The ScanPlant state continuously saved the pose in which the largest fruit part is detected. This pose was set as the visual servo start pose after the plant was fully scanned, under the assumption that a starting pose with a large fruit visibility from the end-effector would result in a more effective final positioning thereof. If no fruit was found, the state returned to the Ready state. Otherwise the visual servo control loop would commence in the Visual Servo state. Each loop cycle triggered and analysed an image for geometrical features. These features were used to calculate the pose correction vector in the camera frame and when applied to the end-effector, centres the camera with the fruit. For reaching towards the fruit, the end-effector moved along its z-axis with a constant speed of 0.03 ms^{-1}. Depth information was therefore not required. Because the fruit target was placed just outside the workspace of the robot, the fruit position was determined as reached when the arm was fully extended and the camera was centred with the fruit. The state machine then returned to the Ready state.

It is assumed that the control law improves the positioning of the end-effector with regard to the fruit centre and therefore ensures the fruit does not leave the view during the visual servo approach. However, this cannot be excluded and we have yet to implement a feature that either reverts to a last known pose that includes a view on a fruit or resumes the plant scanning state.

All states after the Ready state could transition to an Error state. For example, in the Home and ScanPlant state this could occur when the given waypoint pose could not be reached. The Visual Servo state returned to the Error state when no fruit was found during scanning. In the Error state each error could be handled depending on the transition. In this implementation it automatically returned to the Ready state whilst prompting the cause of failure.

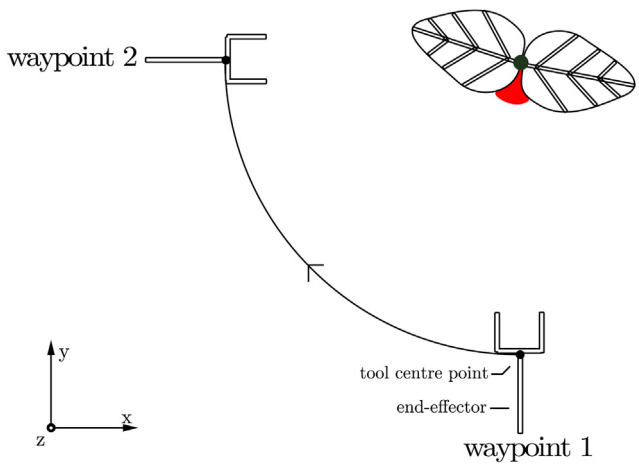


Fig. 8 – Top view of the experimental setup during the ScanPlant state. The end-effector starts at waypoint 1 and follows a trajectory towards waypoint 2 whilst facing the plant and only moves in the horizontal plane *y,x*.

4. Results

The framework was implemented and executed for a dense sweet-pepper crop use case. Software source code of the implementation of the framework is available under the BSD License at the repository found at: https://github.com/rbrth/framework.

The execution of the application resulted in the robot (i) scanning the plant for fruit and (ii) a movement towards the centre of a fruit. A video example can be found at: http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001. In Fig. 9 the last frame of this video is displayed, showing the fruit detection segmentation in the top left and the relative depth estimation from the SLAM node at the bottom left. On the right, the final pose of the end-effector is shown, centred with the camera towards the fruit. The execution time of a single successful execution of all states (excluding the error state) was approximately 45 s.

Supplementary video related to this article can be found at http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001.

4.1. *Plant scanning*

During the plant scanning state, the fruit detection node continuously analysed images from the end-effector. The achieved rate of analysis was 20 Hz, equal to the image acquisition exposure time. In most cases the visited waypoints provided sufficient viewpoints to find a suitable starting location for the visual servo control, meaning that a surface of the fruit was found.

4.2. *Servo control*

During the visual servo control the fruit became more visible, often entirely. The end pose of the end-effector was always centred with the fruit, except for the instances in which the inverse kinematics solver of the robot failed to find a solution of the joint positions. These occasions were characterised by the robot arm already being fully extended to its limits, but not yet horizontally or vertically aligned with the fruit. In Section 5 the cause and solutions to this phenomenon will be discussed.

In a case where a small patch of the fruit surface was made visible from all viewpoints in the horizontal plane of the fruit,

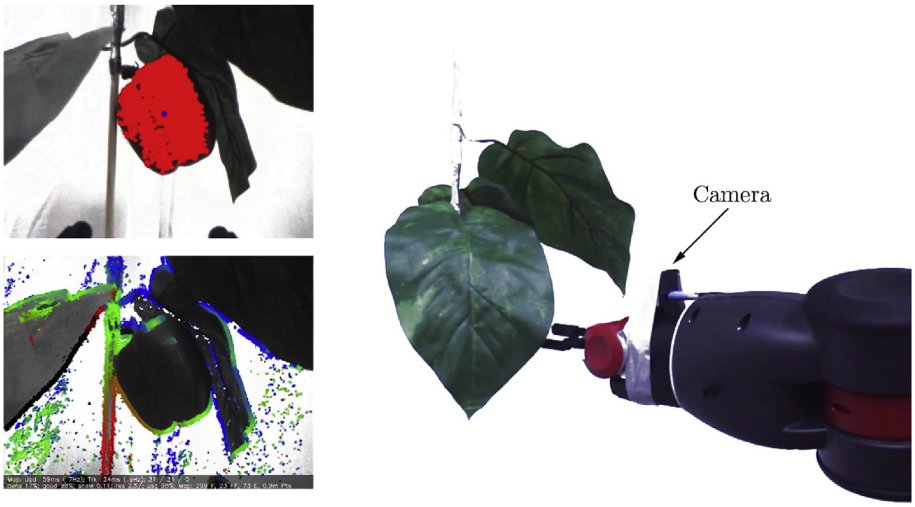


Fig. 9 – Last frame of the video at http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001. The centre of the camera on the end-effector is aligned with the centre of the fruit. The top left image shows the fruit detection node's segmentation. The bottom left image shows the LSD-SLAM node's relative depth estimation. The background in the movie was removed for clarity.

the servo control moved the end-effector in the vertical plane to centre whilst revealing more fruit surface. The result was a curved motion trajectory towards the centre of the fruit. When plant dislocation by a robot entering the crop was simulated by moving the fruit within view of the camera during visual servo control, the algorithm corrected the end-effector accordingly by following the centre of the fruit.

4.3. *Simultaneous localisation and mapping*

During all motions of the robot, the SLAM node ran in parallel to obtain three-dimensional information of the scene and a current estimated pose of the camera. The average publishing rate of respectively new keyframes and pose estimation was on average 5 and 10 Hz As shown in Fig. 10, depth estimations are primarily found on edges in the image. This result is native to the LSD-SLAM library because it operates on image intensity differences. In Section 5 the usability of this result will be discussed.

The bridge node merged and aligned multiple pointclouds from keyframes of the LSD-SLAM node, as displayed in Fig. 11. Again object edges are most discernible, relative depth information within objects without high texture gradients is sparsely available.

5. Discussion

The primary aim of this research was to design a framework for eye-in-hand sensing and motion control to facilitate the development of new robotic harvest applications, especially in dense crops. On a low level, many stand-alone and functionally dedicated libraries are already available to solve parts of this challenge, e.g. ROS, ViSP and LSD-SLAM. Our framework coherently integrates these parts to provide a higher level functional implementation. It can replace custom solutions by providing a standardised approach that supports a variety of use cases. Flexibility is achieved through separation of functional concerns in different modules (Felix & Ortin, 2014). One of the key aspects of the framework design was to add a high degree of implementation flexibility to meet specific use-case constraints. The secondary aim of our

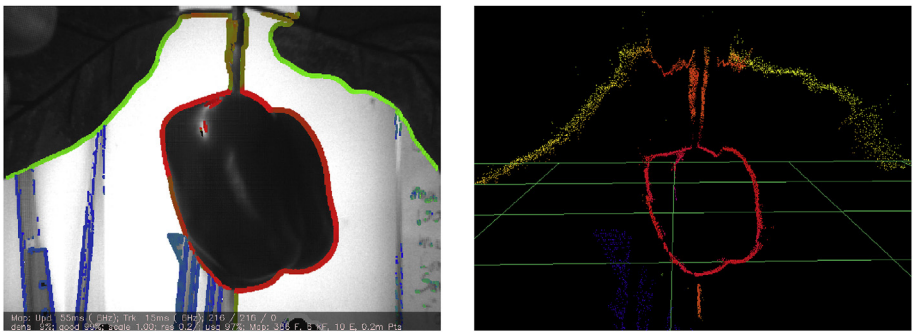


Fig. 10 – The left figure displays a monochrome image with a coloured relative depth estimation overlay from the SLAM node. The figure on the right displays the respective keyframe's pointcloud in ROS RVIZ. The colour gradient indicates relative depth from the end-effector.

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

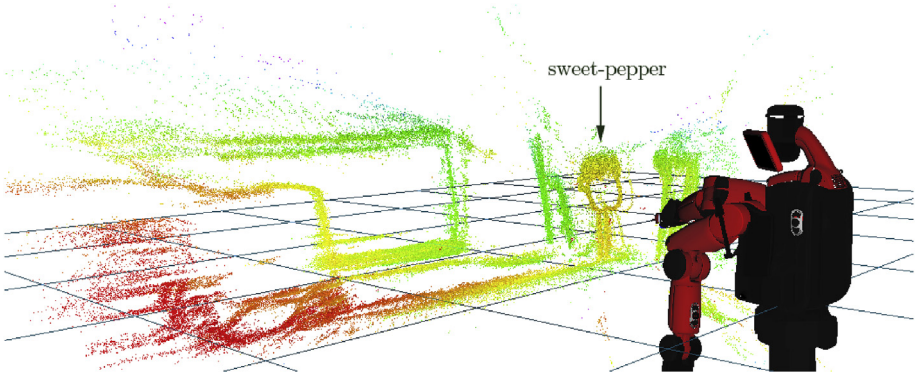


Fig. 11 – Three-dimensional scene reconstruction. Multiple LSD-SLAM keyframes from different perspectives from the camera on the end-effector were merged in a pointcloud. Visualised in ROS RVIZ with Baxter robot pose. An artificial sweet-pepper is placed in front of the end-effector, indicated by the arrow. Other structures in the background can be recognized by their edges. The colour gradient indicates relative depth from the end-effector.

research was to demonstrate a framework implementation for a dense sweet-pepper crop use-case. The example implementation shows the framework can be applied for a sweet-pepper use-case with custom hardware. The added sensing functionality of 3D scene reconstruction shows the extensibility of the framework, without interference with the original software. Results also indicate that the framework can be effective for solving sensing and robot motion control in a dense crop based on visual information from the end-effector, although this should be further explored in a quantitative study under greenhouse conditions.

Other approaches for solving robot sensing and motion control are not always suitable for agri- and horticulture applications due to dense crop vegetation. For sensing fruit, the occlusions of other plant parts can result in false negatives when only a single viewpoint is used (Hemming et al., 2014b; Van Henten et al., 2002). Furthermore, using multiple poses during harvest attempts increases success rates (Henten et al., 2003). Our framework allows for the acquisition of multiple viewpoints by using the motion of the robot in combination with an eye-in-hand approach. During the motion control towards the fruit, corrections may be required when the robot displaces a target after interacting with the dense crop. It is hypothesised that additional viewpoints during motion towards the target can resolve these problems, as well as provide more detailed information of the target. Eye-in-hand sensing is preferred as the perspective from the end-effector is likely to face the target. Some approaches already devote one or more discrete look-and-move actions to correct for displaced targets or refine rough estimated target positions (Hayashi et al., 2010; Van Henten et al., 2002). The approach implemented in our framework uses continuous corrective actions through visual servo control, thereby enabling more corrections and more target information.

On an abstract level, the design of the framework provided a software architecture guideline to approach robot harvest system implementations. The key element of this design was to separate functional concerns to enable modularity, resulting in a system that facilitates extensions and replacements. For example, substituting a camera only affected a single node and could be achieved by replacing a single line of code without the need of recompilation. Adding a node that analyses shared resources does not require other nodes to be changed. Expanding on previous research (Barth et al., 2014; Hellstrom & Ringdahl, 2013), our implementation of this abstract level was done in the ROS middleware, which is innately modular but it does not separate concerns, or functions, automatically. Here each concern was assigned to an individual ROS node, e.g. the concern of coordination that in itself implements a modular FSM. Thus the implementation of the framework remained dependent on ROS and Linux and therefore the flexibility and usability was constrained. For ROS developers this framework is most interesting, for others it provides a useful abstract architectural design guideline. Although modules were functionally separated, they were not fully functionally independent. A notable example is the fruit detection node that was time constrained by the visual servo node. Such dependencies should be identified and avoided, for example by distributing the computation for visual servo control (Wu et al., 2010). The framework was extended with a SLAM node for depth estimation and 3D reconstruction. The nature of the LSD-SLAM algorithm is to look for differences in image intensities, therefore finding matches at texture edges depending on scene contrasts. For optimal scene reconstruction stationary scenes are needed, scene reconstruction should therefore only be used during stationary scenes, e.g. during plant scanning. Further optimisation of scene reconstruction could be implemented by pointcloud registration methods (Rusu & Cousins, 2011). Our framework provides a guideline and implementation for robotic harvest applications that enables access to visual servo control and real-time sensing in a coherent and flexible approach. The relevance of the framework should be confirmed in other use-cases, under real conditions and with further extensions.

The framework was successfully implemented for a sweet-pepper use case and tested under controlled laboratory conditions. Results showed that the application was able to scan an occluded crop for fruit and move the end-effector towards the detected centre of a part or whole of the fruit. A single geometric feature for the servo control algorithm proved sufficient to

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

centre the end-effector and reach the fruit. Furthermore, no depth information was required for a successful servo motion. This indicates that a simplistic and straightforward approach can solve the motion challenge. However, relative depth estimation or descriptive image features, like fruit size, may be required to determine whether the centre of the fruit has been reached. Another possible method includes an air pressure sensor in the suction cup of the end-effector, which triggers upon fruit contact as described in (Hemming et al., 2014c). The execution time per harvest cycle can be improved. Although the Baxter robot has a maximum speed of 0.6 ms^{-1}, a high speed resulted in oversteering during our experiments. When the goal was reached, the spring kinematics of the joints dampen the movement, which resulted in brief imprecise positioning. In a visual servo control loop, consecutive over- and understeering therefore produces an increasing spatial oscillation. Decreasing the speed in our experiments resulted in a more accurate movement with no oversteer, eliminating oscillations. For a real world application, it is suggested that a robot with more precise and faster joint controls is required. The use case implementation and experiments showed that eye-in-hand sensing and motion control is a viable approach for robotic harvesting of a dense crop like sweet-pepper and cucumber. To further validate the use case implementation, a more advanced study under real greenhouse conditions is suggested.

Although no quantitative data was collected, the qualitative performance of the visual servo control library indicated that a more advanced study under real greenhouse conditions is viable. Whilst the artificial crop setup provided a good reflection of the occlusion problems faced in everyday practice, it remains a simplification of the real crop situation, as occlusions from stems and fruit clusters were not taken into account. Nonetheless the framework implementation showed that where a small patch was made visible, the robot managed to find the fruit and move the end-effector towards the fruit centre. This indicates that our approach can be effective under real greenhouse conditions.

The framework can potentially be implemented for robotic harvest use-cases in greenhouses like sweet-pepper, cucumber, tomato or strawberry, or in a more agricultural setting of for instance apple, citrus or broccoli. The feature of using multiple sensing viewpoints is especially functional for dense crops, as multiple viewpoints may be required for fruit detection. Although a distinction can be made between hard and soft obstacles (Bac, Hemming, & Van Henten, 2013), where the former (e.g stems) must be avoided at all costs but the latter (e.g. leaves) can be displaced by the robot to a certain extent, the use of visual servo control may be restricted by the presence of obstacles. Our use-case was successfully implemented using a single geometric point as feature for the visual servo control. For use-cases that require a fixed end-pose, multiple geometric features can be used. However, this requires absolute depth information to model the interaction matrix in Eq. (1). The LSD-SLAM library provides relative depth information because it does not know the scale of the image. To obtain absolute depth information with this library, a calibration procedure can be performed during the initial start of the application by scanning a structure with known dimensions. In the current growing practice crops are frequently revisited to harvest newly ripened fruit, a possible future extension of this research could therefore be to use scene reconstruction to create a world model. The resulting model could be used to i) retry failed harvesting approaches, ii) skip sensing at harvested points, iii) use a crop growth model to extrapolate the position of ripened fruit and iv) update the model. For creating a world model, reconstruction could be limited to relatively stationary plant parts (e.g. fruit and stems) as opposed to frequently moving parts (e.g. leaves that follow the sun). For creating a subset reconstruction, plant part segmentations could be used (Bac, Hemming, & Van Henten, 2014).

5.1. *Conclusion*

The significance of low level libraries that are available to the robot research community to share and build upon common functionality is evident, but individual libraries tackle only isolated concerns, e.g. ROS as communication middleware or ViSP for visual servo control algorithms. At a higher level, a framework can combine these building blocks coherently to provide an orderly structure and a new dimension of utility for a specific set of use-cases. The aim of our research was to provide such a framework for solving two key issues in robot harvesting applications. The first was sensing in dense crops with high fruit occlusions, which requires multiple viewpoints to lower the number of false fruit detection positives. The second was the motion execution using a visual feedback loop. Implementation of this framework for a sweet-pepper use case with dense vegetation indicated viability of the framework and provided insights for further development. Future research should focus on testing the framework under real greenhouse conditions, different use-cases and by extending on functionality.

Acknowledgement

We gratefully acknowledge the support of Lianne H. Scholtens, Joris IJselmuiden and Silke Hemming for the manuscript reviews and helpful comments. Furthermore, we would like to thank the team at RightHand Robotics; Leif Jentoft, Yaroslav Tenzer and Lael Odhner for their insights given during the research. We would also like to thank all other affiliates at the Harvard Biorobotics Laboratory and WUR Greenhouse Horticulture for providing valuable input. Special thanks go to Rob Howe, Douglas Perrin and Pierre Frederic Villard. This research was partially funded by the European Commission in the 7th Framework Programme (CROPS GA no. 246252), in the Horizon2020 Programme (SWEEPER GA no. 644313) and by the Dutch horticultural product board (PT no. 14555).

REFERENCES

Bac, C. W. (2015). *Improving obstacle awareness for robotic harvesting of sweet-pepper* (Ph.D. thesis). Wageningen: Wageningen University & Research Centre.

Bac, C. W., Hemming, J., & Van Henten, E. J. (2013). Robust pixel-based classification of obstacles for robotic harvesting of

sweet-pepper. *Computers and Electronics in Agriculture, 96*, 148–162.
Bac, C. W., Hemming, J., & Van Henten, E. J. (2014). Stem localization of sweet-pepper plants using the support wire as a visual cue. *Computers and Electronics in Agriculture, 105*, 111–120.
Bachiller, M., Cerrada, J., & Cerrada, C. (2003). A modular scheme for controller design and performance evaluation in 3d visual servoing. *Journal of Intelligent and Robotic Systems, 36*, 235–264.
Bachiller, M., Cerrada, C., & Cerrada, J. A. (2006). Designing and building controllers for 3D visual servoing applications under a modular scheme. *Industrial Robotics: Programming, simulation and applications*. InTech.
Baeten, J., Donn, K., Boedrij, S., Beckers, W., & Claesen, E. (2008). Autonomous fruit picking machine: a robotic apple harvester. In C. Laugier, & R. Siegwart (Eds.), *Field and service robotics* (pp. 531–539). Berlin Heidelberg: Springer. volume 42 of Springer Tracts in Advanced Robotics.
Barth, R., Baur, J., Buschmann, T., Edan, Y., Hellstrom, T., Nguyen, T., et al. (2014). Using ros for agricultural robotics : design considerations and experiences. In *Proceeding of the International Conference on Robotics and associated High-technologies and Equipment for Agriculture and forestry (RHEA)* (pp. 509–518).
Bohren, J., & Cousins, S. (2010). The smach high-level executive. *IEEE Robotics and Automation Magazine, 17*, 18–20.
Çalli, B. (2015). *Active grasp synthesis for grasping unknown objects* (Ph.D. thesis). Delft: Delft University of Technology.
Çalli, B., Wisse, M., & Jonker, P. (2011). Grasping of unknown objects via curvature maximization using active vision. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on* (pp. 995–1001).
Chaumette, F. (1998). Potential problems of stability and convergence in image-based and position-based visual servoing. In D. Kriegman, G. Hager, & A. Morse (Eds.), *The confluence of vision and control* (pp. 66–78). London: Springer. volume 237 of Lecture Notes in Control and Information Sciences.
Culjak, I., Abram, D., Pribanic, T., Dzapo, H., & Cifrek, M. (2012). A brief introduction to opencv. In *MIPRO, 2012 Proceedings of the 35th International Convention* (pp. 1725–1730).
De-An, Z., Jidong, L., Wei, J., Ying, Z., & Yu, C. (2011). Design and control of an apple harvesting robot. *Biosystems Engineering, 110*, 112–122.
Dean, T. L., & Wellman, M. P. (1991). *Planning and control. The Morgan Kaufmann series in representation and reasoning*. Los Altos, Calif.: M. Kaufmann Publishers.
Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous localization and mapping: part i. *Robotics Automation Magazine, IEEE, 13*, 99–110.
Engel, J., Schps, T., & Cremers, D. (2014). Lsd-slam: large-scale direct monocular slam. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision ECCV 2014* (pp. 834–849). Springer International Publishing. volume 8690 of Lecture Notes in Computer Science.
Espiau, B., Chaumette, F., & Rives, P. (1992). A new approach to visual servoing in robotics. *Robotics and Automation, IEEE Transactions on, 8*, 313–326.
Felix, J., & Ortin, F. (2014). Aspect-oriented programming to improve modularity of object-oriented applications. *Journal of Software, 9*.
Fitzgerald, C. (2013). Developing baxter. In *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on* (pp. 1–6).
Forssen, P.-E. (2007). Maximally stable colour regions for recognition and matching. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on* (pp. 1–8).
Han, K.-S., Kim, S.-C., Lee, Y.-B., Kim, S.-C., Im, D.-H., Choi, H.-K., et al. (2012). Strawberry harvesting robot for bench-type cultivation. *Biosystems Engineering, 37*, 65–74.
Hayashi, S., Shigematsu, K., Yamamoto, S., Kobayashi, K., Kohno, Y., Kamata, J., et al. (2010). Evaluation of a strawberry-harvesting robot in a field test. *Biosystems Engineering, 105*, 160–171.
Hellstrom, T., & Ringdahl, O. (2013). A software framework for agricultural and forestry robots. *Industrial Robot: An International Journal, 40*, 20–26.
Hemming, J., Bac, C. W., Van Tuijl, B., Barth, R., Bontsema, J., Pekkeriet, E. J., et al. (2014a). A robot for harvesting sweet-pepper in greenhouses. In *Proceedings of the International Conference of Agricultural Engineering*.
Hemming, J., Ruizendaal, J., Hofstee, J. W., & Van Henten, E. J. (2014b). Fruit detectability analysis for different camera positions in sweet-pepper. *Sensors, 14*, 6032–6044.
Hemming, J., Van Tuijl, B., Gauchel, W., & Wais, E. (2014c). Field test of differend end-effectors for robotic harvesting of sweet-pepper. In *Proceedings of the the 29th International Horticultural Congress*.
Henten, E. V., Tuijl, B. V., Hemming, J., Kornet, J., Bontsema, J., & Os, E. V. (2003). Field test of an autonomous cucumber picking robot. *Biosystems Engineering, 86*, 305–313.
Hutchinson, S., Hager, G., & Corke, P. (1996). A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on, 12*, 651–670.
Jara, C. A., Pomares, J., Candelas, F. A., & Torres, F. (2014). Control framework for Dexterous manipulation using dynamic visual servoing and tactile sensors' feedback. *Sensors (Basel, Switzerland), 14*, 1787–1804.
Kitamura, S., & Oka, K. (2005). Recognition and cutting system of sweet pepper for picking robot in greenhouse horticulture. In *Mechatronics and Automation, 2005 IEEE International Conference* (Vol. 4, pp. 1807–1812).
Kröger, T. (2010). *On-line trajectory generation in robotic systems volume 58 of Springer tracts in advanced robotics*. Berlin, Heidelberg, Germany: Springer.
Mahony, R. (2011). Modular design of image based visual servo control for dynamic mechanical systems. In *Proceedings of International Symposium on robotics research* (pp. 1–16). Springer.
Marchand, E. (1999). Visp: a software environment for eye-in-hand visual servoing. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation* (Vol. 4, pp. 3224–3229).
Marchand, E., Spindler, F., & Chaumette, F. (2005). Visp for visual servoing: a generic software platform with a wide class of robot control skills. *Robotics Automation Magazine, IEEE, 12*, 40–52.
Mehta, S., & Burks, T. (2014). Vision-based control of robotic manipulator for citrus harvesting. *Computers and Electronics in Agriculture, 102*, 146–158.
Murphy, R. R. (2000). *Introduction to AI Robotics* (1st ed.). Cambridge, MA, USA: MIT Press.
MVTec Software GmbH. (2015). *Halcon*.
Nguyen, H., Ciocarlie, M., Hsiao, K., & Kemp, C. (2013). Ros commander: flexible behavior creation for home robots. In *ICRA*.
Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T. B., Leibs, J., et al. (2009). ROS: an open-source robot operating system. In *ICRA Workshop on open source software*.
Rusu, R., & Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on* (pp. 1–4).
Song, Y., Glasbey, C., Horgan, G., Polder, G., Dieleman, J., & van der Heijden, G. (2014). Automatic fruit recognition and counting from multiple images. *Biosystems Engineering, 118*, 203–215.
Van Henten, E. J., Hemming, J., Van Tuijl, B., Kornet, J., Meuleman, J., Bontsema, J., et al. (2002). An autonomous robot

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001

for harvesting cucumbers in greenhouses. *Autonomous Robots, 13*, 241–258.

Van Tuijl, B., Wais, E., & Yael, E. (2013). Methodological design of an end-effector for a horticultural robot. In *Proceedings of 4th Israeli Conference on Robotics*.

Wu, H., Lou, L., Chen, C.-C., Hirche, S., & Kuhnlenz, K. (2010). A framework of networked visual servo control system with distributed computation. In *Control Automation Robotics Vision (ICARCV), 2010 11th International Conference on* (pp. 1466–1471).

Yazicioglu, A., Çalli, B., & Unel, M. (2009). Image based visual servoing using algebraic curves applied to shape alignment. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on* (pp. 5444–5449).

Please cite this article in press as: Barth, R., et al., Design of an eye-in-hand sensing and servo control framework for harvesting robotics in dense vegetation, *Biosystems Engineering* (2016), http://dx.doi.org/10.1016/j.biosystemseng.2015.12.001