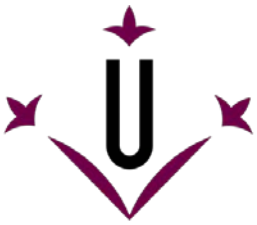

Flexible System of Multiple RGB-D Sensors for Measuring and Classifying Fruits in Agri-food Industry

Rodrigo Méndez Perez[a], Fernando Auat Cheein[a,*], Joan R. Rosell-Polo[b]

[a]*Department of Electronic Engineering, Universidad Técnica Federico Santa María, Valparaíso, Chile*
[b]*Research Group in AgroICT & Precision Agriculture Department of Agricultural and Forest Engineering ETSEA, University of Lleida-Agrotecnio Center, Lleida, Spain*

Abstract

The productivity of the agri-food sector experiences continuous and growing challenges that make the use of innovative technologies to maintain and even improve their competitiveness a priority. In this context, this paper presents the foundations and validation of a flexible and portable system capable of obtaining 3D measurements and classifying objects based on color and depth images taken from multiple Kinect v1 sensors. The developed system is applied to the selection and classification of fruits, a common activity in the agri-food industry. Being able to obtain complete and accurate information of the environment, as it integrates the depth information obtained from multiple sensors, this system is capable of self-location and self-calibration of the sensors to then start detecting, classifying and measuring fruits in real time. Unlike other systems that use specific set-up or need a previous calibration, it does not require a predetermined positioning of the sensors, so that it can be adapted to different scenarios. The characterization process considers: classification of fruits, estimation of its volume and the number of assets per each kind of fruit. A requirement for the system is that each sensor must partially share its field of view with at least another sensor. The sensors localize themselves by estimating the rotation and translation matrices that allow to transform the coordinate system of one sensor to the

*Corresponding author

Email addresses: rodrigo.mendez.11@alumnos.usm.cl (Rodrigo Méndez Perez), fernando.auat@usm.cl (Fernando Auat Cheein), jr.rosell@eagrof.udl.cat (Joan R. Rosell-Polo)

other. To achieve this, Iterative Closest Point (ICP) algorithm is used and subsequently validated with a 6 degree of freedom KUKA robotic arm. Also, a method is implemented to estimate the movement of objects based on the Kalman Filter. A relevant contribution of this work is the detailed analysis and propagation of the errors that affect both the proposed methods and hardware. To determine the performance of the proposed system the passage of different types of fruits on a conveyor belt is emulated by a mobile robot carrying a surface where the fruits were placed. Both the perimeter and volume are measured and classified according to the type of fruit. The system was able to distinguish and classify the 95% of fruits and to estimate their volume with a 85% of accuracy in worst cases (fruits whose shape is not symmetrical) and 94% of accuracy in best cases (fruits whose shape is more symmetrical), showing that the proposed approach can become a useful tool in the agri-food industry.

Keywords: Fruit detection, depth sensor, fruit classification, phenotyping

1. Introduction

Although LiDARs (Light Detection and Ranging) have been widely used in new agricultural technology and applications, the information they provide is associated with distance only. Thus geometric characterization is possible (Andújar et al. (2013)) but the processing of valuable vegetative information is difficult to be further enchanced. In this context, artificial vision systems (such as monocular or stereo cameras, as well as NIR –near infra-red cameras–) are used for monitoring groves (Chéné et al. (2012), Nissimov et al. (2015), Gongal et al. (2015)) and its growing (Gongal et al. (2015), Mehta and Burks (2014)). In particular, we can find nowdays vision systems for unharvested fruit recognition (Schöler and Steinhage (2015), Jay et al. (2015), Xu and Payandeh (2015)), leaf density estimation (Erdal et al. (2015)) and flowers detection and classification (Rosell-Polo et al. (2015)), among other tasks. It is to be noted that within the artificial vision field, light structured sensors are the current research focus in many academic groups, such as (Rosell-Polo et al. (2015)). Light structured sensors (LSS, such as the commercial *Kinect* made by Microsoft Corporation, Redmond, WA, USA) are low cost sensors whose usability in the agronomic context is still under study, as can be seen in a previous work of the authors (Rosell-Polo et al. (2015)). LSS sensors can be used to estimate foliage density, flower density, geometric characteristics

of the orchard or the stems and even terrain parameters (Andújar et al. (2013)), using the depth readings and RGB images provided by the sensor (for this reason they are also called RGB-D or depth sensors). However, the use of LSS or RGB-D sensors in the agri-food industry still is an open issue to be addressed. In this work, we explore the possibility of implementing multiple RGB-D sensors in order to take advantage of the color and depth information.

Multiple vision based solutions have been proposed to solve growing characterization problems. In Jay et al. (2015) the authors developed a system capable of generating a 3D model of plants and classify the different types of plants by analysing their leaves. To achieve this, a mechanical architecture is used, where a color camera is mounted on a metal girder and takes multiple captures from different angles. A similar solution is presented in Yeh et al. (2013), where two color cameras are mounted in a robotic arm, which moves around a leafy vegetable taking multiple pictures. These solutions and others which implement multiple RGB-D sensors proposed to solve similar problems in other areas (Susanto et al. (2012), Satta et al. (2013), Caon et al. (2011), among many others) use specific set-up or need a previous calibration in order to operate.

To overcome the above mentioned issues, in this work a flexible and portable system of multiple RGB-D sensors capable of self location and self calibration of the sensors to then start detecting, classifying and measuring fruits applicable to the agri-food industry is proposed and validated. It is considered the case where different types of fruits are transported in a conveyor belt at the same time. To obtain an accurate classification and characterization, we use computer vision and advanced soft computing methods, which are explained in detail in the following sections. The characterization process considers: classification of fruits, estimation of its volume and the number of assets per each kind of fruit. The entire system works in real-time, with a sampling time of 0.1 seconds, and does not need an expert operator to install it. Processing times is a key issue to face when working in conveyor belts, in order to avoid missing transported fruits or misclassifying them.

This paper is organized as follows. Section 2 presents the proposed system. In Section 3 the implementation and validation of the system and its methods are described. Lastly, Section 4 draws conclusions and provides the guidelines for further work.

2. Materials and Methods

According to the requirements stated in Section 1, the proposed system must be capable of integrating the information acquired from multiple RGB-D sensors without knowing their exact position and orientation, and once the position and orientation of the sensors have been estimated, the RGB-D system should be able to detect, classify and measure the characteristics of the fruits that pass through. The following sections describe in detail the system developed in this work.

2.1. System Architecture

The framework of the proposed system is illustrated in Fig. 1. The first stage deals with the self localization of the sensors, whereas the second stage faces the processing steps to detect, classify and measure the fruits. Briefly,

- We use multiple RGB-D sensors randomly placed over a fruit table. Such sensors correspond to the Kinect v1, manufactured by Microsoft.
- Next, we solve the localization problem: the main goal is to be able to place the Kinect sensors in the work place without increasing the costs of the system, i.e., avoiding further calibration. Then, we study the sensor errors and the error propagation associated with our goal.
- Finally, we analyse the processing stage: RGB and depth information are used to detect and classify fruits.

Following, each part of the system architecture shown in Fig. 1 is presented in detail.

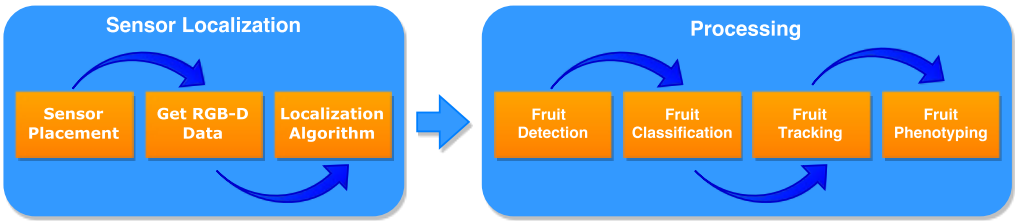

Figure 1: Framework of the proposed work.

2.2. RGB-D Sensor

The RGB-D sensor used is the Microsoft Kinect v1. It can obtain dense depth estimates using a structured light pattern. The device contains a colour camera, an active infra-red camera and a laser projector. The RGB-D sensor uses an infra-red structured random light pattern and interferences will occur if two or more sensors point to the same area. In order to avoid such interference it is possible to alternate the laser projectors (by switching them on and off) and obtaining depth images alternately.

As any other RGB-D sensor, the dephts obtained by the Kinect from Microsoft are affected by measurement errors, which have been widely studied (Andersen et al. (2012)), (Khoshelham (2012)), (Langmann et al. (2012)). The minimum distance that it is able to measure is about ∼800 mm and the maximum distance is about ∼4000 mm.

2.3. Localization

As it was mentioned earlier, a requirement for the system to be able to localize the sensors is that each sensor must share (partially) its field of view with at least another sensor. The sensors localize themselves by estimating the rotation and translation matrices that allow to transform the coordinate system of one sensor into the other. To achieve this the *Iterative Closest Point* (ICP) algorithm (Besl and McKay (1992)), which is capable of estimating the rotation and translation between two point clouds, is used. Since we are able to obtain a depth image from every sensor, it is possible to transform them to point clouds and compute the rotation (R) and translation (T) between two sensors that share part of their fields of view as:

$$\begin{aligned} [R,T] &= ICP(X_i, X_j) \\ X_{j(k)} &\approx RX_{i(k)} + T \quad \forall\, k \;\in\; [1,M] \end{aligned} \tag{1}$$

where $X_i \in \Re^{3\times M}$ corresponds to the point cloud captured by the sensor i, $X_j \in \Re^{3\times M}$ is the point cloud captured by the sensor j and k is a point that belongs to the point cloud, which is composed of M points. $R \in \Re^{3\times 3}$ and $T \in \Re^3$ are respectively the rotation matrix and the translation matrix computed by the ICP algorithm. This process must be performed between all the sensors that share part of their field of view, in order to obtain all the rotation and translation matrices that allow to transform a point cloud from one sensor view to any other.

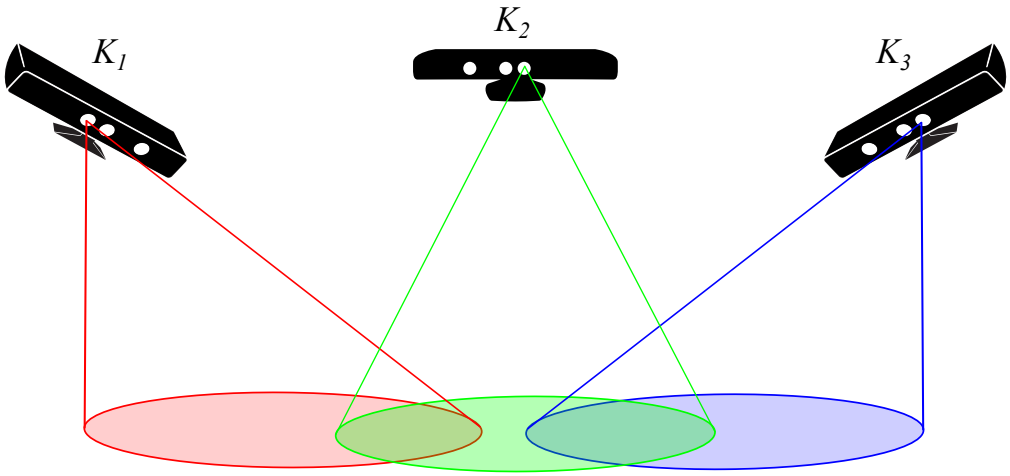

Figure 2: RGB-D Sensors sharing field of view.

If two sensors do not share partially their fields of view, then a third sensor can be used to link both fields of view, as shown in Fig. 2, obtaining the following:

$$\begin{aligned} [R_{1,2}, T_{1,2}] &= ICP(X_1, X_2) \\ [R_{2,3}, T_{2,3}] &= ICP(X_2, X_3) \\ X_1 &\approx R_{2,1}(R_{3,2}X_3 + T_{3,2}) + T_{2,1} \end{aligned} \tag{2}$$

where $R_{i,j}$ is the rotation matrix from point cloud j to point cloud i; identically for $T_{i,j}$. The transformation computed by the ICP algorithm is not exact, since it is an iterative algorithm which converges monotonously to the closest local minimum of the sum of the distance of both point clouds.

2.4. Processing

Once the rotation and translation matrices have been estimated, the processing to detect, classify, measure and track fruits can start. Since this system is meant to work for fruit measuring and classifying in the agri-food industry, there are some assumptions that should be taken into account. It is worth mentioning that all processing stages presented herein were implemented in C/C++ under Windows operating system and using Point Cloud Library (pointclouds.org/) when necessary. In order to ensure real-time performance of the system, the programmed hardware received the maximum

priority from the operating system. We used two computers, one per each Kinect sensor, equipped with processors *Intel Core i5*.

2.4.1. Fruit Detection

Figure 3 shows a representation of the type of situation that our proposed system will have to face.

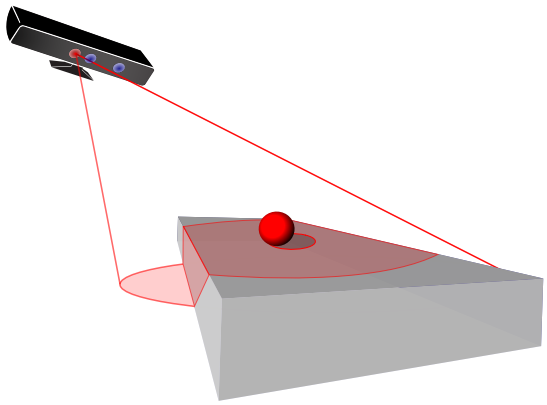

Figure 3: Representation of typical situation for object detection.

The Kinect sensor provides of a depth image, which is then transformed into a point cloud according to the sensor reference system. Later, based on the assumption that the object will be standing on a flat surface, it is possible to find such flat surface by performing a linear fit to the points of the region where the flat surface is located (this needs to be done only once per each sensor). Once the surface is detected, it is necessary to identify all the points that are over this surface, which can be achieved by applying the *Connected Components* algorithm (Samet and Tamminen (1988)), which groups the points that are adjacent to other points. Since our system is meant to detect fruits, it is possible to use thresholds in the number of points and a minimum fitting error to a geometrical primitive that fits best to the 2D projected shape of the fruit, to discriminate whether the points correspond to the fruit of interest or not.

In the case that two or more fruits are detected together as a group by the *Connected Components* algorithm (because they might stand too close from each other), we used the *K-Means* clustering algorithm to reinforce such detection.

2.4.2. Fruit Classification

Once the fruits have been detected it is possible to classify them by extracting different features from the data available. From the RGB image it is possible to extract the colour information of the fruit, whilst the depth image allows the geometrical features of the fruits to be obtained. Since it is necessary to differentiate between multiple types of fruits, a multi-class classifier is preferred. We decided to use a *Multi Layer Perceptron*, as presented in Song et al. (2014), which is a neural network capable of generating a multi-class classifier.

2.4.3. Fruit Measurements

From the depth points that belong to the surface of the detected fruit it is possible to measure geometrical characteristics of the fruit, such as its perimeter, curvature or volume (in this last case some assumptions need to be made since only a part of the surface is detected).

2.5. Fruit Tracking

Assuming that the speed of the table, v, that is transporting the fruits is known and that we are able to get the time between captures of the Kinect sensors, then it is possible to estimate the position x of the object that was previously detected. Thus, let $x^1_{t_0}$ be the fruits detected by the first Kinect at time instant t_0; and $x^2_{t_1}$ the fruits detected by the second Kinect at time t_1 (as stated previously, Kinect sensors do not work simultaneously to avoid interference). Then we can estimate the position of fruits detected by the first Kinect at time t_1 by using the following expression:

$$x^1_{t_1} = x^1_{t_0} + v \times (t_1 - t_0)$$

where t_0 is the time when the previous sensor captured RGB-D data and t_1 is the actual time. Then we can match the detected fruits in $x^2_{t_1}$ with $x^1_{t_1}$ using a closest point strategy, thus allowing us to track the fruits. The latter expression is only valid if $t_1 - t_0$ is relatively small.

2.6. Error Propagation

The proposed methods and hardware used in this work are subject to errors. There are sources of error in the localization of the sensors, the depth measurements performed by the RGB-D sensors, and also as a consequence of the movement of the fruits which is supposed to be linear but this assumption

may not be fulfilled due to the geometry of the fruits or imperfections in the conveyor belt. In this section we analyse such errors and how they propagate. The analysis will be done for two sensors and then the case for multiple sensors will be introduced.

2.6.1. Sources of Errors in the System

Let us consider two depth sensors, K_1 and K_2, positioned in such a way that they share part of their field of view. It is possible to use the coordinate system of K_1 as a global reference system and use the rotation and translation obtained from the ICP algorithm to take the points captured by the sensor K_2 and transform them to the global reference system attached at K_1.

Let $X \in \Re^{3\times M}$ be the set of points that represent the surface of the object of interest in the reference system of the sensor K_1. Let $X_1^{'} \in \Re^{3\times m_1}$ the set of points obtained from the sensor K_1 that describes the surface of the object. Since the set of points captured by the depth sensor has an error in its measurement, we have the following:

$$X_{1(i)}^{'} = X_{1(i)} + \xi_{k_1,x_1} \quad \forall\ i\ \in\ [1, m_1]$$

where $X_1 \in \Re^{3\times m_1}$ corresponds to the distance of the part of the surface captured by the sensor to the reference system of the sensor K_1 and $X_1 \subset X$ $(m_1 < M)$. The error of the depth measurement is represented by $\xi_{k_1,x_1} \in \Re^3$, which is assumed to be a random variable with normal distribution and covariance matrix $\Sigma_{k1,x1} \in \Re^{3\times 3}$.

Assuming that the objects will move with a relatively constant speed over the conveyor belt, we can represent such motion as follows:

$$\overline{X} = \frac{1}{M}\sum_{i}^{M} X_{(i)}$$
$$\overline{X}(t_1) = \overline{X}(t_0) + v(t_1 - t_0) + \eta(t_1)$$

Where $\eta \in \Re^3$ corresponds to a random variable whose distribution can be approximated by a normal distribution with covariance matrix $\Sigma_\eta \in \Re^{3\times 3}$. $\overline{X} \in \Re^3$ is the position of the object, calculated by the mean of all the points that represent the surface of the object.

Let $X_2^{'} \in \Re^{3\times m_2}$ be the set of points that describe the surface of the object, captured by the sensor K_2, similar to $X_1^{'}$, we have the following:

$$X_{2(i)}^{'} = X_{2(i)} + \xi_{k_2,x_2} \quad \forall\ i\ \in\ [1, m_2]$$

where $X_2 \in \Re^{3\times m_2}(m_2 < M)$ corresponds to the real distance of the part of the surface captured by the sensor K_2 in its reference system. Similar to ξ_{k_1,x_1}, ξ_{k_2,x_2} can be approximated to a random variable with normal distribution and covariance matrix $\Sigma_{k2,x2} \in \Re^{3\times 3}$.

Finally, let X_2^R be the set of points X_2 rotated and translated from the reference system of the sensor K_2 to the reference system of the sensor K_1:

$$X_{2(i)}^R = R_{2,1}X_{2(i)} + T_{2,1} \quad \forall\ i\ \in\ [1, m_2]$$

where $R_{2,1} \in \Re^{3\times 3}$ is the rotation matrix and $T_{2,1} \in \Re^3$ is the translation matrix that transform the set of points from the reference system of the sensor K_2 to the reference system of the sensor K_1.

2.6.2. Error in ICP Algorithm

As it was mentioned before, the rotation matrix R and the translation matrix T are estimated by the ICP algorithm, which is an iterative algorithm that tries to reduce the distance between two set of points, and depending on different parameters (initial conditions, number of iterations and threshold on the error), the values of R and T can vary. The error that is introduced by this algorithm can be separated by an error in the rotation matrix and another one in the translation matrix as shown below.

$$\begin{aligned} R &= R^{'} + \Delta_R \\ T &= T^{'} + \Delta_T \\ X_{1(i)} &= (R^{'} + \Delta_R)X_{2(i)} + (T^{'} + \Delta_T) \quad \forall\ i \in\ [1, n] \end{aligned}$$

where $R^{'}$ and $T^{'}$ are the rotation and translation matrix calculated by the ICP algorithm which differ from R and T (real rotation and translation that transform the coordinate system of one RGB-D sensor to the coordinate of another RGB-D sensor) by Δ_R and Δ_T.

2.6.3. Propagation of the different errors

First of all, since our main interest is the position of the fruits and, as mentioned before, it will be calculated as the average of the points that describe the surface of the fruit, if we include the error in the depth measurements,

the position is described as follows:

$$\begin{aligned}\overline{X}^{'} &= \frac{1}{M}\sum_{i}^{M} X_{(i)}^{'} \\ &= \frac{1}{M}\sum_{i}^{M} X_{(i)} + \frac{1}{M}\sum_{i}^{M} \xi_{k_1,x_1(i)} \\ &= E[X] + E[\xi_{k_1,x_1}] \\ &= E[X]\end{aligned}$$

where E is the expectation. Referring now to the error of the ICP algorithm, it is possible to express it as follows:

$$Y_{(i)} = (R^{'} + \Delta_R)X_{(i)} + (T^{'} + \Delta_T) \quad \forall\, i \in\ [1, m_2]$$

$$\begin{pmatrix} y_{1(i)} \\ y_{2(i)} \\ y_{3(i)} \end{pmatrix} = \begin{pmatrix} r_{11} + \Delta_{R_{11}} & r_{12} + \Delta_{R_{12}} & r_{13} + \Delta_{R_{13}} \\ r_{21} + \Delta_{R_{21}} & r_{22} + \Delta_{R_{22}} & r_{23} + \Delta_{R_{23}} \\ r_{31} + \Delta_{R_{31}} & r_{32} + \Delta_{R_{32}} & r_{33} + \Delta_{R_{33}} \end{pmatrix} \begin{pmatrix} x_{1(i)} \\ x_{2(i)} \\ x_{3(i)} \end{pmatrix} + \begin{pmatrix} t_1 + \Delta_{T_1} \\ t_2 + \Delta_{T_2} \\ t_3 + \Delta_{T_3} \end{pmatrix}$$

where $Y = X_2^R$ and $X = X_2$. Assuming Taylor's propagation error we would have the following:

$$\begin{aligned}\Sigma_Y = E[&(R^{'} + \Delta_R)X + T^{'} + \Delta_T - \mu_Y) \\ &((R^{'} + \Delta_R)X + T^{'} + \Delta_T - \mu_Y)^T]\end{aligned} \tag{3}$$

Where Σ_Y is the covariance matrix of Y and μ_Y correspond to the expected value of Y:

$$\mu_Y = E[(R^{'} + \Delta_R)X + T^{'} + \Delta_T]$$

If we expand Eq. 3, terms like $E[X\Delta_R^T]$ or $E[\Delta_R X^T]$ are obtained, which are not possible to estimate in this case due to the fact that their distributions are not really known, and much less if they are a multiplication of two or more random variables. To face such problem, we use the Taylor's series expansion with up to its first order to avoid terms that have two or more

random variables multiplying each other. Expressing each term of Y with its first order Taylor's series expansion we obtain the following:

$$\begin{aligned}
y_i &= f(X, \Delta_{R_i}, \Delta_{T_i}) \\
&= (r_{i1} + \Delta_{R_{i1}})x_1 + (r_{i2} + \Delta_{R_{i2}})x_2 + (r_{i3} + \Delta_{R_{i3}})x_3 + (t_i + \Delta_{T_i}) \\
y_i &\approx f(\hat{X}, \hat{\Delta_{R_i}}, \hat{\Delta_{T_i}}) + \left(\frac{\delta f}{\delta X}\Big|_P\right)^T (X - \hat{X}) \\
&\quad + \left(\frac{\delta f}{\delta \Delta_{R_i}}\Big|_P\right)^T (\Delta_{R_i} - \hat{\Delta_{R_i}}) + \left(\frac{\delta f}{\delta \Delta_{T_i}}\Big|_P\right)^T (\Delta_{T_i} - \hat{\Delta_{T_i}})
\end{aligned}$$

where Δ_{R_i} corresponds to the row i of Δ_R ($\Delta_{R_i} = \begin{pmatrix}\Delta_{R_{i1}} & \Delta_{R_{i2}} & \Delta_{R_{i3}}\end{pmatrix}$) and P is the point where the approximation is made, in this case $P = (\hat{X}, \hat{\Delta_{R_i}}, \hat{\Delta_T})$, where:

$$\begin{aligned}
f(\hat{X}, \hat{\Delta_{R_i}}, \hat{\Delta_{T_i}}) = (r_{i1} + \hat{\Delta_{R_{i1}}})\hat{x}_1 + (r_{i2} + \hat{\Delta_{R_{i2}}})\hat{x}_2 + (r_{i3} + \hat{\Delta_{R_{i3}}})\hat{x}_3 \\
+ (t_i + \hat{\Delta_{T_i}})
\end{aligned}$$

and,

$$\begin{aligned}
\left(\frac{\delta f}{\delta X}\Big|_P\right)^T &= ((r_{i1} + \hat{\Delta_{R_{i1}}}), (r_{i2} + \hat{\Delta_{R_{i2}}}), (r_{i3} + \hat{\Delta_{R_{i3}}})) \\
\left(\frac{\delta f}{\delta \Delta_{R_i}}\Big|_P\right)^T &= (\hat{x}_1, \hat{x}_2, \hat{x}_3) \\
\left(\frac{\delta f}{\delta \Delta_{T_i}}\Big|_P\right)^T &= 1
\end{aligned}$$

If it is assumed that Δ_{R_i} and Δ_T are random variables with normal distribution and covariance matrix $\Sigma_{\Delta_{R_i}} \in \Re^{3\times 3}$ and $\Sigma_{\Delta_{T_i}} \in \Re$ respectively, then,

$$y_i = r_{i1}x_1 + r_{i2}x_2 + r_{i3}x_3 + t_1 + \Delta_{R_{i1}}\hat{x}_1 + \Delta_{R_{i2}}\hat{x}_2 + \Delta_{R_{i3}}\hat{x}_3 + \Delta_{T_i} \quad \forall\ i\ \in\ [1, 3]$$

Rearranging the above expression into a matrix like equation we obtain the following:

$$Y_{(i)} = F(X_{(i)}, \Delta_R, \Delta_T) = R^{'}X_{(i)} + T^{'} + \Delta_R\hat{X}_{(i)} + \Delta_T \quad \forall\ i \in\ [1, n]$$

where, in order to obtain the error propagation, it becomes necessary to modify the structure of some of the matrices, thus to allow multiplications, obtaining the following:

$$\Delta_R = \begin{pmatrix} \Delta_{R_{11}} \\ \Delta_{R_{12}} \\ \Delta_{R_{13}} \\ \Delta_{R_{21}} \\ \Delta_{R_{22}} \\ \Delta_{R_{23}} \\ \Delta_{R_{31}} \\ \Delta_{R_{32}} \\ \Delta_{R_{33}} \end{pmatrix} \qquad \hat{X}_r = \begin{pmatrix} \hat{x}_1 & \hat{x}_2 & \hat{x}_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \hat{x}_1 & \hat{x}_2 & \hat{x}_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \hat{x}_1 & \hat{x}_2 & \hat{x}_3 \end{pmatrix}$$

Then the previous expression changes as follows:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \hat{X}_r \Delta_R + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} + \begin{pmatrix} \Delta_{T_1} \\ \Delta_{T_2} \\ \Delta_{T_3} \end{pmatrix}$$

Finally, we obtain the following:

$$Y = R^{'} X + \hat{X}_r \Delta_R + T^{'} + \Delta_T \tag{4}$$

where its expected value is $\hat{Y}$ and its covariance matrix is Σ_Y:

$$\hat{Y} = R^{'} \hat{X} + T^{'}$$
$$\Sigma_Y = R \Sigma_X R^T + \hat{X}_r \Sigma_{\Delta_R} \hat{X}_r^T + \Sigma_{\Delta_T}$$

where $\Sigma_X \in \Re^{3\times3}$ is the covariance matrix of the points captured by the sensor K_2, $\Sigma_{\Delta_R} \in \Re^{9\times9}$ is the covariance matrix of the rotation matrix R and $\Sigma_{\Delta_T} \in \Re^{3\times3}$ is the covariance matrix of the translation matrix.

Thus, we have obtained the error propagation expression for two sensors.

2.6.4. Error Propagation for Multiple Kinects

Similar to what was obtained above, we now consider the case that was presented in Fig. 2 (three Kinect sensors) and Eq. 2. If we consider the error in the ICP algorithm it will be as follows:

$$X_3^{R1} = (R_{2,1} + \Delta_{R_{2,1}})((R_{3,2} + \Delta_{R_{3,2}})X_3 + T_{3,2} + \Delta_{T_{3,2}}) + T_{2,1} + \Delta_{T_{2,1}}$$

where it is possible to use the approximation calculated before in Eq. 4, obtaining the following:

$$
\begin{aligned}
X_3^{R1} &= F(F(X_3, R_{3,2}, T_{3,2}), R_{2,1}, T_{2,1}) \\
X_3^{R1} &= F(R_{3,2}X_3 + \hat{X}_{3r}\Delta_{R_{3,2}} + T_{3,2} + \Delta_{3,2}, R_{2,1}, T_{2,1}) \\
X_3^{R1} &= R_{2,1}R_{3,2}X_3 + R_{2,1}\hat{X}_{3R}\Delta_{R_{3,2}} + R_{2,1}T_{3,2} + R_{2,1}\Delta_{T_{3,2}} \\
&\quad + E[R_{3,2}X_3 + \hat{X}_{3R} + T_{3,2} + \Delta_{T_{3,2}}]\Delta_{R_{2,1}} + T_{2,1} + \Delta_{T_{2,1}} \\
X_3^{R1} &= R_{2,1}R_{3,2}X_3 + R_{2,1}\hat{X}_{3R}\Delta_{R_{3,2}} + R_{2,1}T_{3,2} + R_{2,1}\Delta_{T_{3,2}} \\
&\quad + (R_{3,2}\hat{X}_3 + T_{3,2})\Delta_{R_{2,1}} + T_{2,1} + \Delta_{T_{2,1}}
\end{aligned}
$$

As it was done before, there is again a dimensionality issue to be faced since the term $(R_{3,2}\hat{X}_3 + T_{3,2}) \in \Re^3$ is multiplied by $\Delta_{R_{2,1}} \in \Re^9$. This can be done assuming the following matrices:

$$
I_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad
I_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad
I_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
$$

197 Therefore,

$$
\begin{aligned}
\hat{X}_r = F_I(X) = I_1 E[X] \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} + I_2 E[X] \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \\
+ I_3 E[X] \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}
\end{aligned}
$$

Then the expected value and covariance matrix of X_3^{R1} are the following:

$$
\begin{aligned}
E[X_3^{R1}] &= R_{2,1}R_{3,2}\hat{X}_3 + R_{2,1}T_{3,2} + T_{2,1} \\
\Sigma_{X_3^{R1}} &= R_{2,1}R_{3,2}\Sigma_{X_3}R_{3,2}^T R_{2,1}^T + R_{2,1}\hat{X}_{3r}\Sigma_{\Delta_{R_{3,2}}}\hat{X}_{3r}^T R_{2,1}^T \\
&\quad + \Sigma_{\Delta_{T_{2,1}}} + F_I(R_{3,2}\hat{X}_3 + T_{3,2})\Sigma_{\Delta_{R_{2,1}}}F_I(R_{3,2}\hat{X}_3 + T_{3,2})^T \\
&\quad + R_{2,1}\Sigma_{\Delta_{T_{3,2}}}R_{2,1}^T
\end{aligned}
$$

Since $\Delta_{R_{3,2}}$, $\Delta_{R_{2,1}}$, $\Delta_{T_{3,2}}$ and $\Delta_{T_{2,1}}$ are the errors of the ICP algorithm, they can be considered to have the same second moment (i.e., rotations have the same covariance matrix and translations have the same covariance matrix). Then the expression for X_3^{R1} corresponds to the following:

$$\begin{aligned}\Sigma_{X_3^{R1}} = R_{2,1}R_{3,2}\Sigma_{X_3}R_{3,2}^T R_{2,1}^T + R_{2,1}\hat{X}_{3r}\Sigma_{\Delta_R}\hat{X}_{3r}^T R_{2,1}^T + R_{2,1}\Sigma_{\Delta_T}R_{2,1}^T \\ + F_I((R_{3,2}\hat{X}_3 + T_{3,2})\Sigma_{\Delta_R}F_I((R_{3,2}\hat{X}_3 + T_{3,2})^T + \Sigma_{\Delta_T}\end{aligned}$$

Thus, we have obtained above the expression for the error propagation for three sensors. If more sensors are to be used, then the above expression should be obtained in the manner it was shown in this section.

2.7. Integration of Measurements

To manage the errors and thus to improve the fruits detection and classification a Kalman Filter is used, since it has a process model which involves an unknown variable (in this case the position of the fruit) and observations (measurements of part of the surface of the fruits from multiple sensors). Following, we derive the expressions of the Kalman Filter as was implemented in this work.

2.7.1. Prediction

The prediction of the position of each fruit is possible to be performed at every time instant, where the value of the unknown variable is estimated with the process model, which in this case corresponds to the movement of the fruit in the flat surface with constant speed. The prediction equations of the Kalman Filter are shown below:

$$\begin{aligned}X(t_1) &= X(t_0) + V(t_1 - t_0) + \eta(t_1) \\ \hat{X}(t_1|t_0) &= \hat{X}(t_0|t_0) + V(t_1 - t_0) \\ \Sigma_X(t_1|t_0) &= \Sigma_X(t_0|t_0) + \Sigma_\eta\end{aligned}$$

where X is the position of the fruit and Σ_X its covariance matrix, V is the constant speed of what would be the conveyor belt, $\eta(t)$ is a Gaussian noise with covariance Σ_η, and $\hat{X}$ is the expected value of X.

2.7.2. Update

Once a measurement is done, it is possible to update the value of the unknown variable, merging the information obtained by the prediction and

the measurement, as shown below:

$$
\begin{aligned}
Y(t_1) &= RX(t_1) + \hat{X}_r(t_1)\Delta_R + T + \Delta_T \\
Z(t_1) &= Y(t_1) - (R\hat{X}(t_1|t_0) + T) \\
S &= R^{'}\Sigma_X(t_1|t_0)R^{'T} + \hat{X}_r(t_1|t_0)\Sigma_{\Delta_R}\hat{X}_r(t_1|t_0)^T + \Sigma_{\Delta_T} \\
K(t_1) &= \Sigma_X(t_1|t_0)R^T S^{-1} \\
X(t_1|t_1) &= \hat{X}(t_1|t_0) + K(t_1)Z(t_1) \\
\Sigma_X(t_1|t_1) &= (I - K(t_1)R)\Sigma_X(t_1|t_0)
\end{aligned}
$$

where $Y(t_1)$ is the measurement made, i.e. the position of the same fruit made with another RGB-D sensor and with the rotation matrix (R) and translation matrix (T) calculated with the *ICP* algorithm. Then, $Z(t_1)$ is the difference between the measurement and the predicted value of the measurement. Finally, the expected value and covariance matrix of X is updated with the Kalman gain, $K(t_1)$, and the matrix S, which takes into account the covariance of Δ_R and Δ_T.

2.8. Error Localization

In order to model the error that the ICP algorithm introduces into the system an experiment was performed in which a Kinect was mounted in a KUKA robotic arm. Since the robotic arm has 6 degree of freedom it is possible to place the arm in different positions and capture several shots from different points with great positioning accuracy. To validate this experiment, we use the accurate encoder of the KUKA arm.

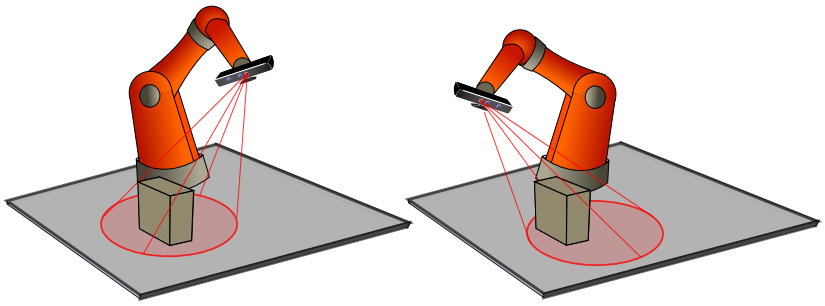

Figure 4: Experiment to estimate ICP error. Two different examples of depth images taken by then Kinect mounted on KUKA robotic arm.

As shown in Fig. 4, the idea is to choose different positions but all of them pointing to the same object. Then, it is possible to use the ICP algorithm between two different depth images to estimate R and T (rotation and translation matrices). Since the position of the arm its known, it is possible to calculate the real rotation and translation between the two different positions of the robotic arm and compare them with the results of the *ICP* algorithm.

Figure 5 shows the Kinect mounted on the KUKA arm and some examples of the depth images that were taken from different positions pointing to the same object.

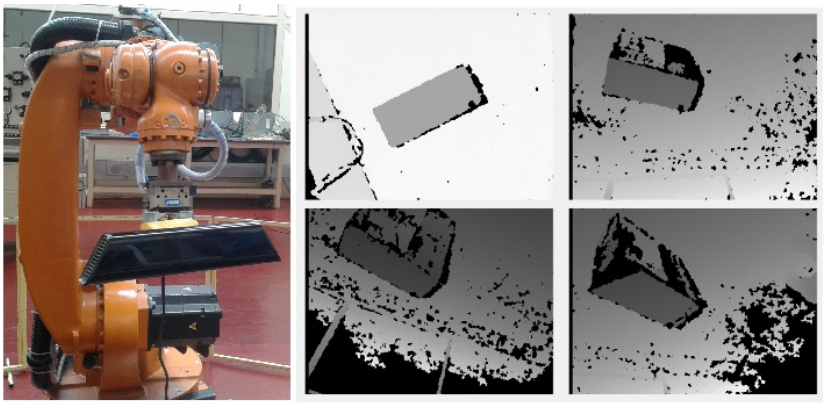

Figure 5: Kinect mounted on KUKA robotic arm and depth images taken from different positions.

Let $R_1 \in \Re^{3\times3}$ and $T_1 \in \Re^3$ the rotation and translation that transform the coordinate system from the base of the robot to the end of the robotic arm in the first position, and $R_2 \in \Re^{3\times3}$ and $T_2 \in \Re^3$ the rotation and translation that transform the coordinate system from the base of the robot to the end of the robotic arm in the second position. Then, considering X_B as the base position, X_1 as the first position and X_2 as the second position, it is possible to obtain the rotation and translation between the two positions by doing the following:

$$\begin{aligned} X_1 &= R_1 X_B + T_1 \\ X_2 &= R_2 X_B + T_1 \\ R_1^{-1}(X_1 - T_1) &= R_2^{-1}(X_2 - T_2) \\ X_1 &= R_1 R_2^{-1} X_2 - R_1 R_2^{-1} T_2 + T_1 \end{aligned}$$

Then, the differences between the rotation and translation obtained by the ICP algorithm (R_{ICP}, T_{ICP}) and the ones obtained by the robotic arm (R_{Real}, T_{Real}) are calculated as shown below:

$$E_R = R_{ICP} - R_{Real}$$
$$E_T = T_{ICP} - T_{Real}$$

An example of the difference between the matching of two point clouds done by the ICP algorithm and by calculating the real rotation and translation is shown in Fig. 6 (red points correspond to one point cloud and green ones to the other point cloud). We can see that both matchings –the one obtained using only the enconders of the robot manipulator (left figure) and the one from the ICP algorithm (right figure)– are visually consistent, but the ICP produces mismatches that could eventually lead to a bad fruit characteri- zation. Therefore, there is a need to know the errors associated with the matching process through the use of the ICP approach.

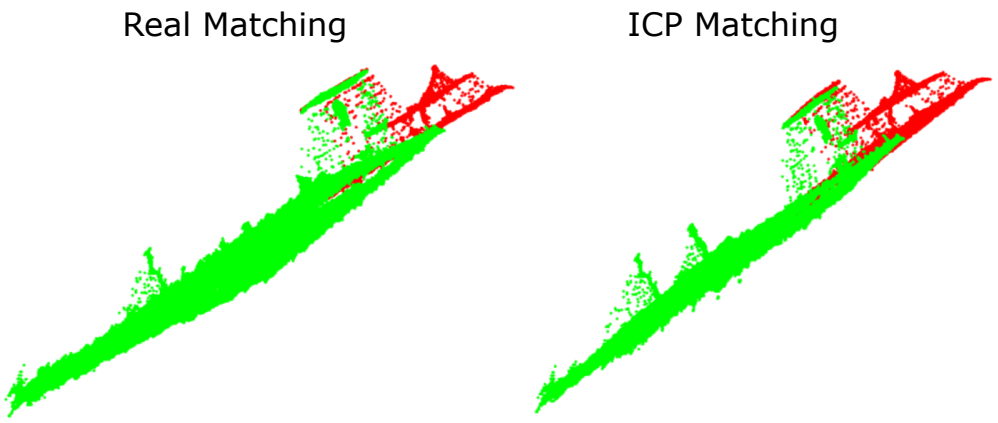

Figure 6: Comparison between matching of two point clouds. Left picture shows the real matching and right picture the matching obtained with the values of R and T estimated by the ICP algorithm

To estimate the error in the rotation and translation matrices, 10 depth images from 10 different positions were taken, making it possible to calculate 90 different rotations and translations. Since it was proposed that the error on the components of the rotation and translation matrices are normally distributed, a Gaussian was fitted to the error. To modelate the error, we

have chosen ten random locations of the camera attached to the end-effector of the robot manipulator. In all cases, the sensor was pointing to the target.

Figure 7 shows the values of the mean and standard deviation obtained by fitting the error to a Gaussian distribution for all the components of the rotation and translation matrices.

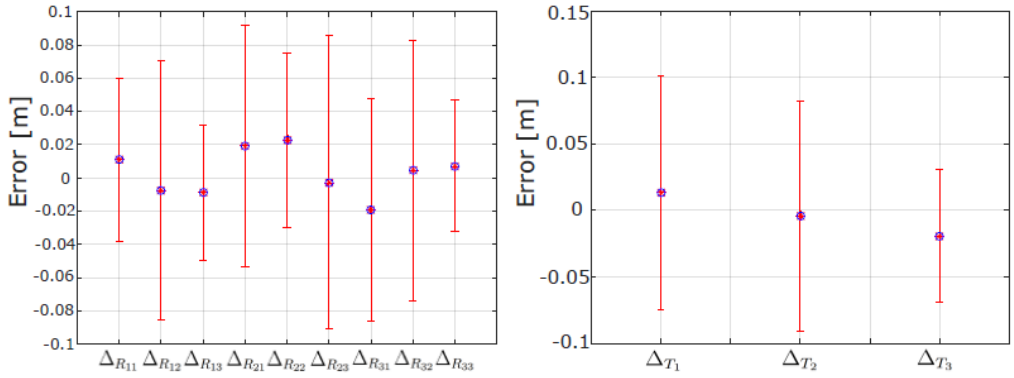

Figure 7: Mean and covariance from Gaussian distribution for R and T errors.

Such covariance matrices are then used for R and T in our system (see Eq. 1).

3. Results

This section is aimed at providing empirical results of the different processes described in this brief, namely: fruit detection and classification, fruit measurement and characterization, fruit tracking and experimental results after the integration of all processes.

3.1. Fruit Detection

As described in Section 2.4.1 the fruits were detected by grouping the points that are above the surface and discarding the group of points which has a high error when fitting to an ellipse. An example is shown in Fig. 8. The left picture shows the depth points captured by the Kinect and the flat surface that was fitted. In Fig. 8, right, it is shown the original depth image and, highlighted in red, the group of pixels that pass the ellipse fit after applying connected components.

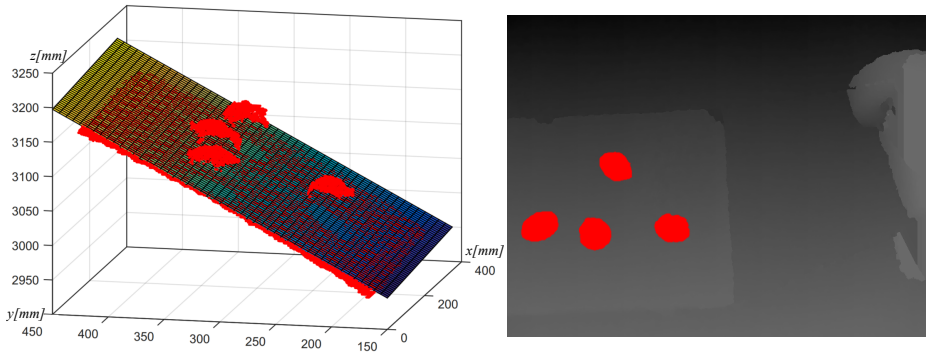

Figure 8: Fruit detection example; Left picture shows the point cloud and the fitted surface and right picture shows, highlighted in red, the pixels that are detected as fruits.

Figure 9 shows an example when two fruits are too close from each other and it is not possible to differentiate them just by finding the points that are over the surface and grouping them with connected components. In this case, if the area of the selected pixels is between two predefined thresholds, the *K-Means* algorithm is applied and iterated from 2 to a maximum number of clusters (in this case we used 5), it stops if the points clustered by *K-Means* have a low error when applying the fit with the ellipse.

Figure 9: Example where K-Means is used to separate two close fruits. Highlighted in red and blue two fruits too close from each other.

In the example there are two fruits which are too close of each other and the *K-Means* algorithm separate them into two groups, the first group, in blue, that corresponds to the green apple and the second group, in red, that corresponds to the red apple.

(a) 5 Red and 6 Green Apples (b) 5 Chilean Avocados (c) 6 Lemons

Figure 10: Classification example of the different tested fruits.

3.2. Fruit Classification

To classify the fruits, five different types of fruits were used: lemon, green apple, red apple, Chilean avocado and Peruvian avocado. The *Multi Layer Perceptron* was trained using 50 samples of each type of fruit. The feature vector that is used as an input is constructed as follows: first, a frame of 40×40 pixels is placed where the fruit is located; then, this section of the color image is transformed from RGB to HSV, in order to make the classification more robust against changes in illumination. Next, a histogram is created for each channel (hue, saturation and brightness), each one with 10 bins, filling it with the information of the image; and finally, the values of the bins of the three histograms are concatenated with the perimeter and volume of the detected fruit (normalized by the volume of the cube that contains the object), forming the feature vector.

Figure 10 shows an example of the classification for the different tested fruits under different light conditions varying from 100 lx to 1000 lx, thus emulating field conditions. In Fig. 10.b there were 5 Chilean avocados but the one in the bottom was misclassified as a Peruvian avocado. Apart from that one all the other fruits were correctly classified.

Figure 11 shows the confusion matrix we obtained with the testing group of fruit samples. Nevertheless, such matrix corresponds to 50 trials totally. It is possible to see that it only gets confused between the Chilean avocado and the Peruvian avocado. This is because their color is very similar and the difference between sizes is not big enough. To obtain the results shown above, and as will be explained in detail in Section 3.5, the fruits were located at a platform carried by a mobile robot, where fruits were moving on the platform

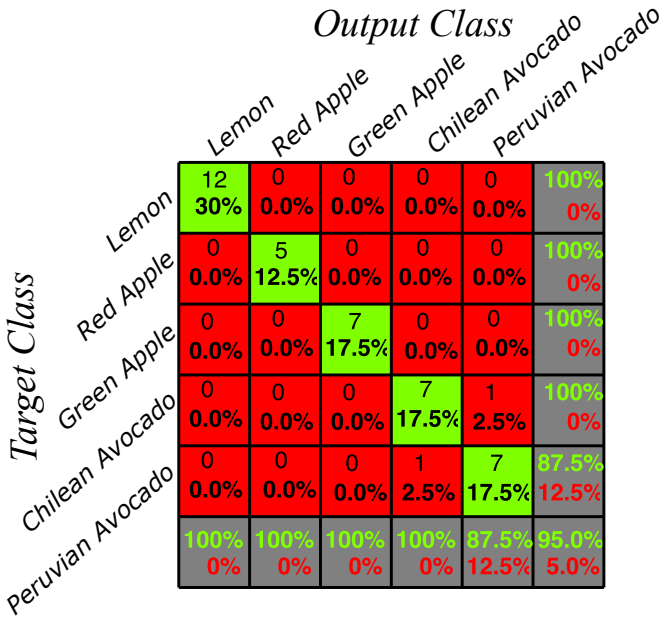

Figure 11: Confusion Matrix of the 5 types of fruits.

due to their inertia to the robot's motion during the trials.

3.3. Fruit Measurements and Characterization

Two parameters were extracted from each fruit: its perimeter and its volume. It is possible to calculate the perimeter of the fruit by using only the points extracted from the depth image, by fitting the contour to an ellipse and then calculating the perimeter.

In the case of the volume, some assumptions need to be made since it is not possible to measure all the points of the surface due to the blind spots that it might have. Since the fruits tested where apples, avocados and lemons, it is possible to assume some level of symmetry. Therefore, in order to calculate the volume of the fruit, it is assumed that the points captured are the half and the other half are symmetrical to the ones that were captured. In Fig. 12 it is shown the result of this method, where the red points are the ones that were obtained by the depth sensor and the green ones are created based on the assumption that the fruit is symmetric.

In Fig. 13 is shown the error in the volume estimation for the different type of fruits tested in this study, compared with the real volume previously measured with an accurate beaker.

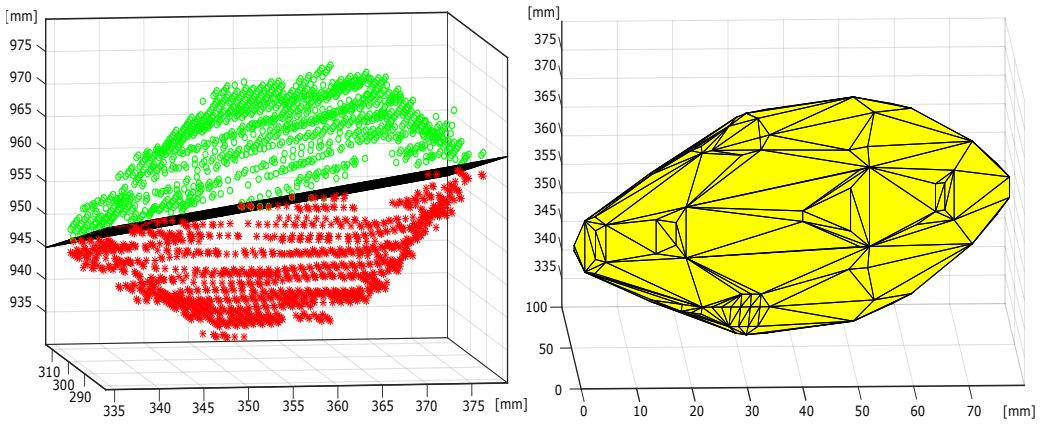

Figure 12: Volume estimation of a lemon. Left picture shows the points measured (in red) and the ones replicated assuming that it is symmetrical (green points) and right picture is the resulting convex hull of the green and red points.

It is possible to see that the error is high in some cases. Nevertheless, since the system characterizes the volume of each fruit every time it is detected per each camera, it then calculates the mean volume, which is the final value thrown by the system to the user.

3.4. Fruit Tracking

To track each fruit, the Kalman Filter is implemented with the values obtained in the experiments described in Section 2.8, shown below:

$$\Sigma_{\Delta_R} = diag \begin{pmatrix} 0.0493 \\ 0.0782 \\ 0.0408 \\ 0.0728 \\ 0.0523 \\ 0.0885 \\ 0.0672 \\ 0.0782 \\ 0.0396 \end{pmatrix} \quad \Sigma_{\Delta_T} = diag \begin{pmatrix} 0.0878 \\ 0.0862 \\ 0.0501 \end{pmatrix} \quad \Sigma_{\eta} = diag \begin{pmatrix} 0.001 \\ 0.001 \\ 0.001 \end{pmatrix}$$

where Σ_η is defined as an error of 1 cm for each component.

The Kalman filter compensates the error and performs the estimation of the position based on the process model and the measurement, but a previous matching needs to be done in order to conclude if a fruit measured in the new

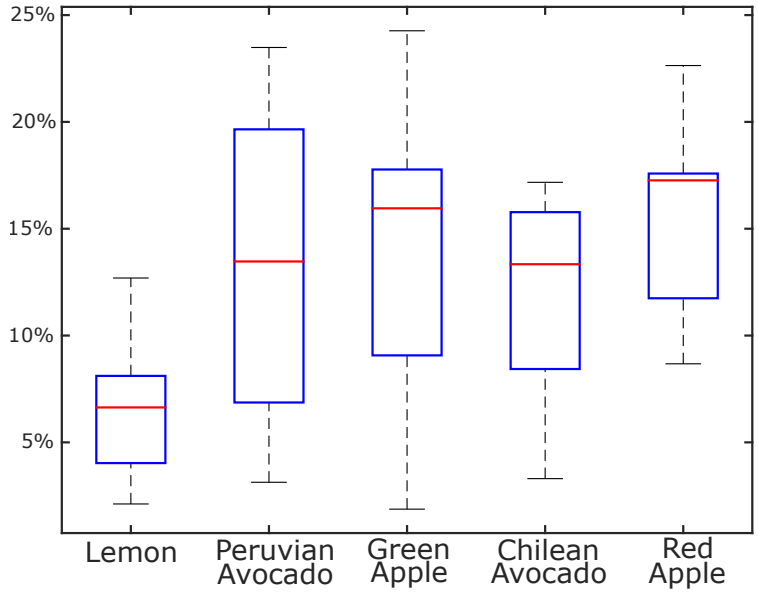

Figure 13: Volume estimation error, in percentage, between the estimated volume and the real volume previously obtained with an accurate beaker.

depth image is actually a new fruit or corresponds to one already measured before. To do this the Mahalannobis distance is used. The Mahalannobis distance, as a matching metric d_m, is shown below.

$$d_m(x) = \sqrt{(x-\mu)S^{-1}(x-\mu)^T}$$

The matching fruit selected is the one with the lowest d_m distance and if the distance calculated is over a threshold previously defined, then it is assumed that it is a new fruit.

3.5. Experimental Results

To test our system and algorithms, two Kinects were mounted on a steel structure, both pointing to the area where the fruits will pass through. To emulate a conveyor belt we programmed a mobile robot to carry a light white surface were the fruits were placed, as shown in Fig. 14. This mobile robot moves at a constant speed of 0.1 m/s, in order to simulate a conveyor belt. The fruits were likely to move due to their inertia to the robot's motion. The obtained results include suche movement of the fruits since they were not attached to such white platform. In addition, each Kinect is connected

to its own computer; one of such computers operates as the main processing system, while the other one gathers the images and does a pre-processing on the depth images.

Figure 14: Experimental Setup. Lef image shows the mobile robot carrying fruits and right image the kinects mounted in a metal structure. The Kinect in the left corresponds to the sensor K_1 and the Kinect in the right correspond to the sensor K_2.

Three different runs were done. In the first trial only lemons were placed; in the second trial green and red apples; and, in the third trial, Chilean avocados and Peruvian avocados as shown in Fig. 15. The trials were repeated ten times, although only one replication is shown here. The remaining trials showed similar results.

Figure 15: Three tests where fruits were detected, classified, tracked and measured. First test was done only with lemons; in the second test green and red apples where placed and the third test Chilean and Peruvian avocados where tested.

To depict the functionality of our system, Fig. 16 shows the tracking done in the test with the lemons. In the first row several depth images taken from the first Kinect (K_1) are shown. Each detected lemon has a number sequentially assigned as they were detected. In addition, we show the covariance ellipse of each lemon's position using the Kalman Filter, but projected to the plane. In the second row we show the depth images taken from the other Kinect (K_2).

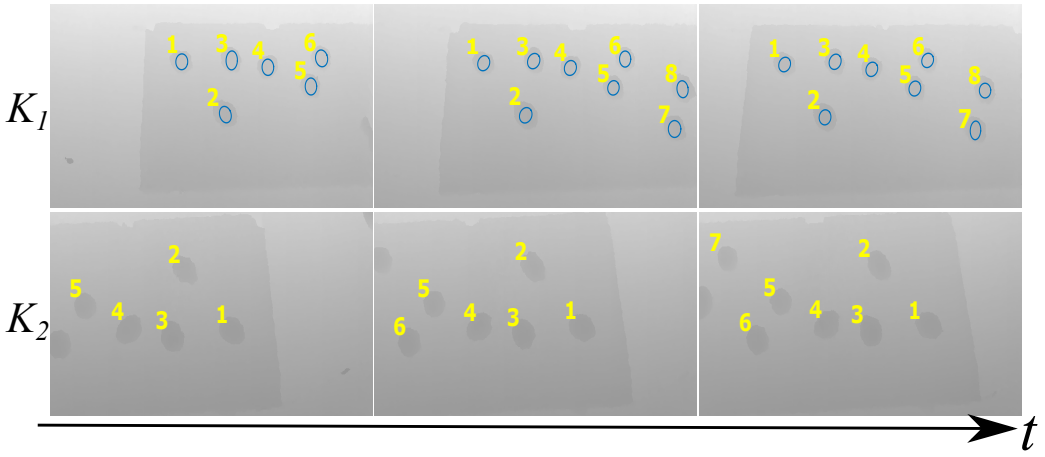

Figure 16: Depth images of test conducted with lemons. Each fruit is tracked and its error is shown with blue ellipse. The bottom line represents time.

Note that in the depth images taken by sensor K_1 (first row in Fig. 16) the surface is moving from right to left, while in the depth images taken by sensor K_2 (second row in Fig. 16) the surface is moving from left to right, since the sensors were placed facing the floor but in opposite direction, as shown in Fig. 14. The previous results show that the system is able to identify and match correctly the fruits that were detected in a previous depth image with the fruits detected in a new depth image.

4. Conclusions

In this work, we presented a portable and flexible system for the agri-food industry. Such system was aimed at classifying and characterizing fruits in a conveyor belt using an arrangement of Kinect sensors that do not need of extra calibration procedures but to partially share their field of view. An ICP algorithm was used for self-positioning of the sensors with respect to each other. However, the positions of the sensors in the industrial environment were treated as random variables. The use of a 6 degree of freedom KUKA robotic arm proved to be valuable for the assessment of the error introduced by the ICP algorithm, in the conditions where this system is pretended to be used. The KUKA arm allowed us to obtain the first and second moments associated with the ICP algorithm when estimating the positions of the Kinect sensors. We used the RGB and depth information provided by

the set of Kinects to classify fruits in a conveyor belt with avocados, lemons and apples, and to characterize them, obtaining their size, volume and thus to estimate the amount of production. The entire system was tested in real time with a mobile robot emulating the conveyor belt, being the synchronization of the sensors and modelling of the error propagation in our system the main challenges. In the experimentation, the system was able to distinguish and classify 95% of fruits and to estimate their volume with to estimate their volume with accuracies up to 85% in worst cases (fruits whose shape is not symmetrical) and 94% in best cases (fruits whose shape is more symmetrical), showing that our approach can become a useful tool in the agri-food industry.

In future works, the authors will test the performance of the system in long term experimentation in the agri-food industry over a real conveyor belt.

5. References

Andersen, M., Jensen, T., Lisouski, P., Mortensen, A., Hansen, M., Gregersen, T., Ahrendt, P., 2012. Kinect Depth Sensor Evaluation for Computer Vision Applications. Electrical and Computer Engineering Technical, 37.
URL http://eng.au.dk/fileadmin/DJF/ENG/PDF-filer/Tekniske_rapporter/Technical_Report_ECE-TR-6-samlet.pdf

Andújar, D., Rueda-Ayala, V., Moreno, H., Rosell-Polo, J. R., Escolà, A., Valero, C., Gerhards, R., Fernández-Quintanilla, C., Dorado, J., Griepentrog, H. W., 2013. Discriminating crop, weeds and soil surface with a terrestrial LIDAR sensor. Sensors (Switzerland) 13 (11), 14662–14675.

Besl, P., McKay, N., 1992. A Method for Registration of 3-D Shapes.

Caon, M., Yue, Y., Tscherring, J., Mugellini, E., Abou Khaled, O., 2011. Context-Aware 3D Gesture Interaction Based on Multiple Kinects. Applied Sciences (c), 7–12.

Chéné, Y., Rousseau, D., Lucidarme, P., Bertheloot, J., Caffier, V., Morel, P., Belin, É., Chapeau-Blondeau, F., 2012. On the use of depth camera for 3D phenotyping of entire plants. Computers and Electronics in Agriculture 82, 122–127.

Erdal, E., Abramov, A., Wörgötter, F., Scharr, H., Fischbach, A., Dellen, B., 2015. Modeling leaf growth of rosette plants using infrared stereo image sequences. COMPUTERS AND ELECTRONICS IN AGRICULTURE 110, 78–90.
URL http://dx.doi.org/10.1016/j.compag.2014.10.020

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., Lewis, K., 2015. Sensors and systems for fruit detection and localization: A review.

Jay, S., Rabatel, G., Hadoux, X., Moura, D., Gorretta, N., 2015. In-field crop row phenotyping from 3D modeling performed using Structure from Motion. COMPUTERS AND ELECTRONICS IN AGRICULTURE 110, 70–77.
URL http://dx.doi.org/10.1016/j.compag.2014.09.021

Khoshelham, K., 2012. Accuracy Analysis of Kinect Depth Data. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII-5 (August), 133–138.

Langmann, B., Hartmann, K., Loffeld, O., 2012. Depth Camera Technology Comparison and Performance Evaluation. Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods, 438–444.
URL http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0003778304380444

Mehta, S. S., Burks, T. F., 2014. Vision-based control of robotic manipulator for citrus harvesting. Computers and Electronics in Agriculture 102, 146–158.
URL http://dx.doi.org/10.1016/j.compag.2014.01.003

Nissimov, S., Goldberger, J., Alchanatis, V., 2015. Obstacle detection in a greenhouse environment using the Kinect sensor. Computers and Electronics in Agriculture 113, 104–115.

Rosell-Polo, J. R., Cheein, F. A., Gregorio, E., Andújar, D., Puigdomènech, L., Masip, J., Escolà, A., 2015. Advances in Structured Light Sensors Applications in Precision Agriculture and Livestock Farming. In: Advances in Agronomy. Vol. 133. pp. 71–112.

Samet, H., Tamminen, M. K., 1988. Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees. IEEE Transactions on Pattern Analysis and Machine Intelligence 10 (4), 579–586.

Satta, R., Pala, F., Fumera, G., Roli, F., 2013. Real-time appearance-based person re-identification over multiple Kinect cameras. ... on Computer Vision Theory ..., 1–4.
URL http://192.167.131.140/sites/default/files/Satta_VISAPP2013.pdf

Schöler, F., Steinhage, V., 2015. Automated 3D reconstruction of grape cluster architecture from sensor data for efficient phenotyping 114, 163–177.

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.
URL http://www.sciencedirect.com/science/article/pii/S1537511013002109

Susanto, W., Rohrbach, M., Schiele, B., 2012. 3D Object Detection With Multiple Kinects. Proc. European Conference on Computer Vision ({ECCV}'2012), 1–10.
URL http://link.springer.com/chapter/10.1007/978-3-642-33868-7_10

Xu, G., Payandeh, S., 2015. Sensitivity study for object reconstruction using a network of time-of-flight depth sensors. Robotics and Automation (ICRA), 2015 IEEE International Conference on, 3335–3340.

Yeh, Y.-h. F., Lai, T.-c., Liu, T.-y., Liu, C.-c., Chung, W.-c., Lin, T.-t., 2013. Special Issue : Image Analysis in Agriculture An automated growth measurement system for leafy vegetables 5. Biosystems Engineering 117, 43–50.
URL http://dx.doi.org/10.1016/j.biosystemseng.2013.08.011