Box-Scan: An efficient and effective algorithm for box dimension measurement in conveyor systems using a single RGB-D camera

Ho-min Park\textsuperscript{ab,*,} Arnout Van Messem\textsuperscript{ac}, Wesley De Neve\textsuperscript{ab}

\textsuperscript{a} Center for Biotech Data Science, Department of Environmental Technology, Food Technology and Molecular Biotechnology, Ghent University Global Campus, Songdo, Incheon, Korea
\textsuperscript{b} IDLab, Department of Electronics and Information Systems, Ghent University, Ghent, Belgium
\textsuperscript{c} Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Ghent, Belgium

* Corresponding Author: Ho-min Park <homin.park@ghent.ac.kr>

Abstract

Different algorithms for object dimension measurement have recently been proposed, leveraging the power of RGB-D cameras. However, most of these algorithms are not suitable for deployment in the logistics industry: they are only able to deal with non-moving objects and/or the usage of multiple cameras introduces time delays. In this paper, we introduce Box-Scan, a novel algorithm that enables real-time box dimension measurement in conveyor systems using a single RGB-D camera. We discuss the industrial setting in which the proposed algorithm needs to operate, as well as a prototype that integrates our algorithm. Furthermore, we provide an analysis of the effectiveness of our prototype as a function of the conveyor speed, demonstrating that the prototype built comes with a maximum measurement error of less than 5% at a conveyor speed of 3.4 km/h.

Keywords: Algorithms, Computer vision, RGB-D camera

1. Introduction

The logistics industry uses boxes for protecting, loading, and packaging a plethora of items. To improve the efficiency of these actions, technology for box dimension measurement can be of high value. When developing measurement technology for logistics use cases, three important requirements have to be taken into account. First, the technology needs to be deployed in a low-speed environment. Indeed, people typically need to intervene at some point in time, resulting in a limitation on the conveyor speed that can be utilized. Third, boxes have been designed to protect their content, allowing for deformations in order to absorb the impact of shocks. In practice, this implies that a certain error tolerance is allowed when performing dimension measurement.

Given the aforementioned requirements, we can argue that the use of high-speed and high-precision equipment for box dimension measurement is not cost effective in a logistics setting. Therefore, in our research and development effort, we have decided to make use of a widely available RGB-D camera for box dimension measurement, and where this type of RGB-D camera is also frequently used in the area of computer vision. Specifically, in this paper, we present Box-Scan, a hierarchical algorithm that can perform box dimension measurement on a conveyor in real time, making use of a single RGB-D camera, a common desktop computer, and some fixing equipment. The proposed algorithm consists of two major steps. The first step deploys a Region-of-Interest (ROI) processing procedure, only extracting those parts from the depth frames of the RGB-D camera that are necessary for performing box dimension measurement, making it possible to significantly reduce the computational complexity. The second step deploys a state control procedure that automatically detects the presence of objects on a conveyor belt, making it possible to for instance transition between an idle state and a measurement state.

For investigating the efficiency and the effectiveness of Box-Scan, we created a prototype using a Microsoft Kinect V2 camera. We tested this prototype using various conveyor systems.
speeds, demonstrating that we can measure box dimensions with a relative error lower than 3 to 6% and an absolute error lower than 10 mm, for a conveyor belt operating at a speed of 3.4 km/h.

2. Preliminaries

2.1 RGB-D cameras and depth frames

Next to basic color information (Red, Green, and Blue; RGB), RGB-D cameras are able to generate distance values called depth (D), mainly through the usage of infrared radiation (IR). Depending on the mode of use, RGB-D cameras can be classified into Time-of-Flight, Structured Light [1], and Active Stereo [2] cameras. Regardless of the mode of use, the user can obtain a depth frame (map) from the camera, complementing the color information.

A depth frame is basically a 2-D matrix, with each element of the matrix storing an estimated distance value. Given the availability of distance values, we can easily convert each 2-D point in a depth frame into a 3-D point in the real world, using a coordinate transformation matrix.

2.2 RGB-D cameras for dimension measurement

RGB-D cameras can acquire both color and depth frames. Since the release of the Kinect for the Xbox 360 in 2010 [3], RGB-D cameras have become widely available at an affordable price.

Diverse research efforts have been conducted using RGB-D cameras. In particular, reconstruction studies that acquire 3-D data by scanning [4]–[9] and research efforts that analyze human behavior and gestures [10]–[14] have become mainstream. Moreover, various research efforts have also focused on performing object dimension measurements [15]–[18]. The approach of Hamamura et al. [17] is of particular interest, focusing on box dimension measurement (width, height, length) using three RGB-D cameras. Specifically, this approach has been tested in an industrial setting, measuring the dimensions of three different types of boxes located on a conveyor belt operating at a speed of 0.9 m/s. The measured dimension values were reported to have a maximum absolute error of 11 mm.

2.3 Random sample consensus

RANdom SAmple Consensus (RANSAC) [19] is an iterative algorithm that estimates model parameters from data typically containing outliers. While the least squares method finds model parameters by minimizing the sum of squared residuals, RANSAC finds model parameters that come with the greatest consensus, excluding outliers that were present in the observed data. In this paper, we use RANSAC to eliminate IR-related outliers (like black pixels and flying points [1]), so to be able to extract more accurate dimension information from the depth frames of the RGB-D camera.

3. Proposed Approach

Let us assume the scenario shown in Figure 1. In particular, we have a conveyor belt rotating at a constant speed, with a 3-D camera (black rectangle) located above the conveyor belt, looking down from the top. The red triangle denotes the detection area and the green dotted line is the Region-of-Interest (ROI) of the RGB-D camera, with an object (cuboid consisting of blue lines) passing through this ROI. Given this scenario, and as shown in Figure 2, our algorithm measures the object dimensions from the time the object touches the

Figure 1 Scenario illustrating real-time box dimension measurement on a moving conveyor belt using a single RGB-D camera (black rectangle).

Figure 2 System flow diagram. The proposed system consists of an RGB-D camera and a computer. The RGB-D camera first captures RGB-D images of the conveyor surface, sending these images to the system frame buffer. Our algorithm retrieves depth frames from the buffer at a speed of 10 to 20 FPS. As soon as an object has completely passed through the ROI, the state control algorithm executes the SaveOutput() procedure and sends the result to the target system. The system subsequently gets a depth frame from the RGB-D camera and determines the three dimensions of the object.
ROI until the time the object has passed completely through
the ROI.

3.1 ROI processing

As shown in Figure 1, a box passes through a triangle-shaped
detection area and the ROI projected by our RGB-D
camera. Upon projection on the surface of the conveyor belt
and the box, this ROI takes the form of a line of points, both
in the horizontal and the vertical direction (see also Figure 4),
and where this line-shaped ROI can be used to perform
dimension measurement.

A point \( p(u, v) \in \text{ROI} \), located on the conveyor surface,
is stable at first. This means that the distance between the
ROI point \( p_t(u, v) \) in the current depth frame \( t \) and the
same ROI point \( p_{t+1}(u, v) \) in the next depth frame \( t + 1 \)
is smaller than \( \epsilon_d \). The threshold \( \epsilon_d \) is used to determine
whether an object is present or absent on the conveyor
surface below the RGB-D camera. If an object passes
through the ROI, some of the points in the ROI will have a
distance difference that is greater than \( \epsilon_d \). We say that these
points are in an excited state. We subsequently define \( s_t \) as
the state of a point at time \( t \):

\[
s_t(u, v) = \begin{cases} 
1 \text{(excited)}, & \text{if } d(p_{t-1}(u, v), p_t(u, v)) > \epsilon_d \\
0 \text{(stable)}, & \text{otherwise}
\end{cases}
\]

In the above equation, \( d \) denotes the Euclidean
distance between two points. Furthermore, we represent the
number of excited-state points in the ROI as \( N_{\text{excited}} \):

\[
N_{\text{excited}}(\text{ROI}) = \sum_{(u, v) \in \text{ROI}} s(u, v) \tag{1}
\]

Assuming that an object passes through the ROI, we can
then leverage Equation (2) to define the set \( E_t \), which
contains the excited-state points, and Equation (3) to define
the set \( S_t \), which contains the stable points, at time \( t \):

\[
E_t = \{ p_t(u, v) \colon s_t(u, v) = 1, p_t(u, v) \in \text{ROI}_t \} \tag{2}
\]

\[
S_t = \{ p_t(u, v) \colon s_t(u, v) = 0, p_t(u, v) \in \text{ROI}_t \} \tag{3}
\]

Furthermore, we introduce the following notation:
\( N(\text{ROI}_t) = |E_t| + |S_t| \). In addition, we represent the
initial set of points by means of \( S_0 \), with this set having the
following property: \( N(\text{ROI}_0) = |S_0| \) (that is, all points in
the ROI are initially stable).

Algorithm 1 ROI-measurement

1: **Procedure** ROI-Measurement()
2: \hspace{1em} height ← 0
3: \hspace{1em} width(length) ← 0
4: \hspace{1em} D ← ∅
5: // height by averaging distances
6: \hspace{1em} for all \( p_{E_t}(u, v), p_{S_t}(u, v) \) where \( p(u, v) \in E_t \) do
7: \hspace{2em} D ← Distance(\( p_{E_t}(u, v), p_{S_t}(u, v) \)) ∪ D
8: \hspace{1em} end for
9: \hspace{1em} height ← (∑_d∈D d) / |D|
10: // width(length) by finding maximum distance within \( E_t \)
11: \hspace{1em} E_r ← \text{RANSAC} \rightarrow \text{LineFitting}(E_t)
12: \hspace{1em} for all \( p_i \in E_r \) do
13: \hspace{2em} \text{end for}
14: \hspace{1em} width(length) ← Max(width(length), Distance(p_i, p_j))
15: \hspace{1em} end for
16: \hspace{1em} \text{end for}
17: \hspace{1em} return width(length), height
18: **end procedure**

19: **Procedure** Distance(p, q)
20: \hspace{1em} Convert 2-D points \( p \) and \( q \) into 3-D points \( p' \) and \( q' \)
21: \hspace{1em} return Euclidean distance between \( p' \) and \( q' \)
22: **end procedure**

Using the aforementioned equations, we present the
ROI-driven measurement of the width and height of a box in
Algorithm 1. As will be further clarified in the next section,
we use the same approach for determining the width and
length of a box.

The procedure **ROI-Measurement** takes as input the set
\( E_t \) of excited-state points and the initial set \( S_0 \) of stable
points, calculating the width (length) and height of a box. The
procedure **Distance** takes as input a point \( p \) from the set
\( E_t \) and a point \( q \) from the set \( S_0 \), returning the 3-D
Euclidean distance between the two given points, after
having converted the 2-D points \( p \) and \( q \) into the 3-D
points \( p \) and \( q \).

For calculating the height of a box, the **ROI-
Measurement** procedure makes use of an average. For
calculating the width (length) of a box, the **ROI-
Measurement** procedure makes use of an iterative approach
to find the distance between the farthest two points,
deploying RANSAC-LineFitting to eliminate outliers when
creating straight lines.
Algorithm 2 State Transition Control

1: procedure State-Control (N_row, N_col)
2: \( S_{row}, S_{col}, E_{row}, E_{col}, H, W, L, ROI_{row}, ROI_{col} \leftarrow \emptyset \)
3: state, excited, height, width, length \leftarrow \emptyset
4: \( \epsilon_N \leftarrow 10, \epsilon_d \leftarrow 0.05 \)
5: while true do
6: \( f \leftarrow getDepthFrame() \)
7: \( ROI_{row}, ROI_{col} \leftarrow \text{Get} \rightarrow \text{ROI}(N_{row}, N_{col}) \)
8: \( \text{excited} \leftarrow N_{\text{excited}}(ROI_{col}) \) (1)
9: if state = 0 then
10: \( S_{row}, S_{col} \leftarrow (3) \)
11: state \leftarrow 1
12: end if
13: if state = 1 then
14: if excited \( > \epsilon_N \) then
15: state \leftarrow 2
16: height, width, length \leftarrow 0
17: end if
18: end if
19: if state = 2 then
20: if excited \( > \epsilon_N \) then
21: \( E_{row}, E_{col} \leftarrow (2) \)
22: height, width \leftarrow ROI \rightarrow \text{Measurement}(E_{row}, S_{row})
23: height, length \leftarrow ROI \rightarrow \text{Measurement}(E_{col}, S_{col})
24: \( H \leftarrow \text{height} \cup \text{H}, W \leftarrow \text{width} \cup \text{W}, L \leftarrow \text{length} \cup \text{L} \)
25: else
26: SaveOutput (H, W, L)
27: state \leftarrow 1
28: end if
29: end if
30: end while
31: end procedure

32: procedure Get-ROI(f, N_{row}, N_{col})
33: \( ROI_{row}, ROI_{col} \leftarrow \emptyset \)
34: \( ROI_{row} \leftarrow \{ (\text{p}(u, v)) | v = N_{col}/2, 0 \leq u < N_{row}, \text{p}(u, v) \in f \} \)
35: \( ROI_{col} \leftarrow \{ (\text{p}(u, v)) | 0 \leq v < N_{row}, \text{p}(u, v) \in f, v = N_{col}/2 \} \)
36: return \( ROI_{row}, ROI_{col} \)
37: end procedure

3.2 State transition control
Table 1 Width estimation (in mm) at a conveyor speed of 3.4–km/h, using 10 samples for each class. All error values are absolute in nature.

<table>
<thead>
<tr>
<th>Box Class</th>
<th>Standard Size</th>
<th>Measured Size after Assembly</th>
<th>Estimated Size</th>
<th>Avg Error</th>
<th>Max Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Absolutes</td>
<td>Relative</td>
</tr>
<tr>
<td>1</td>
<td>190</td>
<td>192</td>
<td>195.09</td>
<td>3.09±2.27</td>
<td>1.61</td>
</tr>
<tr>
<td>2</td>
<td>180</td>
<td>191</td>
<td>193.07</td>
<td>4.29±4.44</td>
<td>1.93</td>
</tr>
<tr>
<td>2-1</td>
<td>250</td>
<td>263</td>
<td>265.98</td>
<td>4.91±2.55</td>
<td>1.19</td>
</tr>
<tr>
<td>3</td>
<td>250</td>
<td>261</td>
<td>265.56</td>
<td>4.71±3.52</td>
<td>1.80</td>
</tr>
<tr>
<td>4</td>
<td>310</td>
<td>322</td>
<td>325.26</td>
<td>5.40±6.49</td>
<td>1.67</td>
</tr>
</tbody>
</table>

Table 2 Height estimation (in mm) at a conveyor speed of 3.4–km/h, using 10 samples for each class. All error values are absolute in nature.

<table>
<thead>
<tr>
<th>Box Class</th>
<th>Standard Size</th>
<th>Measured Size after Assembly</th>
<th>Estimated Size</th>
<th>Avg Error</th>
<th>Max Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Absolutes</td>
<td>Relative</td>
</tr>
<tr>
<td>1</td>
<td>90</td>
<td>92</td>
<td>89.92</td>
<td>2.07±0.44</td>
<td>2.25</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>154</td>
<td>151.83</td>
<td>2.16±0.65</td>
<td>1.40</td>
</tr>
<tr>
<td>2-1</td>
<td>100</td>
<td>106</td>
<td>102.97</td>
<td>3.13±0.56</td>
<td>2.85</td>
</tr>
<tr>
<td>3</td>
<td>210</td>
<td>216</td>
<td>212.58</td>
<td>3.41±0.70</td>
<td>1.57</td>
</tr>
<tr>
<td>4</td>
<td>280</td>
<td>292</td>
<td>288.07</td>
<td>3.92±0.57</td>
<td>1.34</td>
</tr>
</tbody>
</table>

Table 3 Length estimation (in mm) at a conveyor speed of 3.4–km/h, using 10 samples for each class. All error values are absolute in nature.

<table>
<thead>
<tr>
<th>Box Class</th>
<th>Standard Size</th>
<th>Measured Size after Assembly</th>
<th>Estimated Size</th>
<th>Avg Error</th>
<th>Max Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Absolutes</td>
<td>Relative</td>
</tr>
<tr>
<td>1</td>
<td>220</td>
<td>219</td>
<td>221.67</td>
<td>2.76±1.35</td>
<td>1.26</td>
</tr>
<tr>
<td>2</td>
<td>270</td>
<td>273</td>
<td>273.21</td>
<td>4.91±5.99</td>
<td>1.80</td>
</tr>
<tr>
<td>2-1</td>
<td>350</td>
<td>358</td>
<td>360.29</td>
<td>3.02±5.21</td>
<td>1.25</td>
</tr>
<tr>
<td>3</td>
<td>340</td>
<td>338</td>
<td>342.66</td>
<td>4.66±2.94</td>
<td>1.37</td>
</tr>
<tr>
<td>4</td>
<td>410</td>
<td>413</td>
<td>418.34</td>
<td>7.80±8.29</td>
<td>1.90</td>
</tr>
</tbody>
</table>

Figure 5 Relative average error of the dimension measurements as a function of the conveyor speed, with errors being less than 5% at all conveyor speeds. When the conveyor speed exceeds 3.4–km/h, the relative average error of the width and the length increases significantly.
4. Implementation

To test the effectiveness of the proposed algorithm, we created a prototype using a Kinect v2 camera [20] and two computer vision libraries [21], [22]. For our experiments, we also simulated a conveyor environment by making use of a treadmill and several boxes.

4.1 Prototype description

Our prototype consisted of a PC configured with an Intel 3.5GHz i5-4690 CPU, 8GB RAM, and an NVIDIA GeForce GTX 970 GPU. The depth frame rate of the Kinect v2 is known to be 30 frames per second (FPS). However, for our purposes, the depth frame rate was confirmed to be between 10 and 13 FPS, given the calculation times needed by our algorithm. The Kinect v2 was fixed at 1.12 meters above the treadmill using a boom stand. In addition, the camera was installed in such a way that it looked down the treadmill vertically. Since the range of the ROI varies depending on the position of the camera, we adjusted the range for a depth frame in the procedure Get-ROI as follows: ROI\text{col} as \((u = 255,112 \leq v < 312)\) and ROI\text{row} as \((80 \leq u < 430, v = 212)\).

4.2 Prototype visualization

Figure 4 depicts the external appearance and the line-shaped ROIs used by our prototype. In particular, the red triangle delineates the detection area and the green dotted lines represent the ROIs. Since an object is passing through the ROI, we can see that some points on the green dotted lines have been excited. Figure 4(a) shows the ROI in the column direction, whereas Figure 4(b) shows the ROI in the row direction.

Figure 3 depicts RGB images (a, b, c) and depth images (d, e, f), as obtained from the RGB-D camera used. The green lines on the RGB images denote the projected ROIs. In addition, the yellow arrow to the right of each image denotes the direction in which the conveyor is moving. Dimension measurement starts as soon as an object meets the vertical ROI, and as soon as a box has passed completely, the obtained dimension information is sent to SaveOutput(). In the case of a depth frame, the closer the object, the darker the colors. We can observe a number of black regions that are not well measured, due to light reflection and camera noise.

5. Performance evaluation

For our experiments, we used five standard boxes from a post office. The relevant dimension of each box can be found in the second column of Table 1, Table 2, and Table 3. We can confirm we could observe a difference between the advertised dimensions and the real-world dimensions after assembly of the boxes. As a result, we measured each dimension again after assembly of the boxes. The obtained values can be found in the column labeled “Measured Size after Assembly” of the respective tables. Our experiment was performed by varying the conveyor speed from 1.0~km/h to 3.4~km/h. We then analyzed the box dimension values obtained by our algorithm.

We implemented the SaveOutput() procedure in such a way that it estimates the hand-measured dimension values of a particular box by taking the median of each set \(W\) (width), \(H\) (height), and \(L\) (length), thus mitigating the impact of outliers. We denote an estimated dimension value as \(\hat{y}\).

Following our approach, each box was processed ten times and the average of the measured values can be found in the column labeled “Estimated Size” of the respective tables. In addition, we calculated the average absolute error, the

![Image](image-url)

Figure 6 Relative maximum error of the dimension measurements as a function of the conveyor speed, with errors being less than 6% at all conveyor speeds, with the exception of a conveyor speed of 3.8~km/h. When the conveyor speeds exceed 3.4~km/h, the relative maximum error of the width and the length increases significantly.
average relative error, the maximum absolute error, and the maximum relative error. These values are shown in the respective columns of each table. The absolute error \( e_{\text{ABS}} \) and the relative error \( e_{\text{REL}} \) were calculated as \( e_{\text{ABS}} = |y - \hat{y}| \) and \( e_{\text{REL}} = \frac{e_{\text{ABS}}}{\hat{y}} \times 100 \).

Given Table 1, Table 2, and Table 3, we can observe that both the average and the maximum absolute error increase as the box size increases, but the average relative error is always less than 3% and the maximum relative error is always less than 5%, indicating that our prototype is industrially usable.

Furthermore, we set up an experiment to investigate how the relative error varies as a function of the conveyor speed. To that end, we made use of the same experimental settings as before, only adjusting the speed of the conveyor. Figure 5 shows the average relative error as a function of the conveyor speed and Figure 56 shows the maximum relative error as a function of the conveyor speed. We can observe that height measurement is not significantly affected by the conveyor speed used: both the average and maximum relative error are less than 6%. On the contrary, the average and maximum relative errors for the width and the length increase when the conveyor speed increases to 3.8 km/h. In other words, to achieve a maximum relative error of less than 6%, the conveyor speed used should be less than 3.8 km/h. We consider two reasons for this: (1) the conveyor speed is too fast compared to the scan speed in order to obtain sufficient data and (2) the high conveyor speed, used over a short distance, causes the object to vibrate, increasing the error.

Compared to [17], the proposed algorithm is efficient and effective in several ways. First, we tested our algorithm using various conveyor speeds and confirmed that it works well at a speed of 3.4 km/h, which is faster than the conveyor speed used by Hamamura et al. [17] (0.9 m/s = 3.24 km/h).

Second, our approach utilizes only one RGB-D camera, unlike the previous study that combines three RGB-D cameras. That way, we are able to dramatically reduce the computational complexity of our algorithm. Indeed, compared to the 1.5s needed, our prototype is able to generate on-the-fly dimension measurements, producing dimension estimates right after a box passed through the ROI (that is, results are obtained within about 0.1s after a box passed through the ROI).

6. Conclusions and directions for future work

In this paper, we have proposed Box-Scan, a novel box dimension measurement algorithm in support of logistics automation. We presented a two-level algorithm, with the low-level algorithm being responsible for box dimension measurement and the high-level algorithm being responsible for system state management, transitioning between a measurement state and an idle state, depending on the presence or absence of a box on a moving conveyor. We integrated the proposed algorithm into a functional prototype, using commodity hardware: a single RGB-D camera and a desktop PC. Our experimental results demonstrate that our measurement approach can determine the dimensions of a box located on a conveyor belt moving at a speed of up to 3.4 km/h with a maximum relative error of 6%. This result is similar or better than the state-of-the-art, while only using a single RGB-D camera in an operational environment that is more challenging (that is, an environment using a higher conveyor speed). Moreover, the proposed algorithm is able to generate box dimension measurements within about 0.1 seconds after a box has left the region-of-interest monitored by our RGB-D camera.

We can identify two major directions for future research, further improving the proposed algorithm. First, the higher the camera, the larger the area covered by the camera. This allows the algorithm to measure larger objects, but conversely, the area covered by each point is also larger, resulting in a lower accuracy. Therefore, it would be of interest to investigate the relation between accuracy, camera distance, and measurable size limits. Second, the angle of entry of a box. The current scenario assumes that all objects are cuboids and that each side is either parallel or perpendicular to the conveyor plane. In practice, this assumption can be enforced by installing a physical guide on the conveyor, or we can relax this assumption by enhancing the algorithm proposed in this paper.

Acknowledgment

The research and development activities described in this paper were funded by Ghent University Global Campus (GUGC) and MOTIE (Ministry Of Trade, Industry, and Energy), Korea, under the Advanced Technology Center programme supervised by KEIT (Korea Evaluation Institute of Industrial Technology) (10077285, Development of a smart and automated packaging system for variable-type and variable-size products).

References

[1] P. Zanuttigh, G. Marin, C. Dal Mutto, F. Dominio,


