VOLUMNECT - Measuring Volumes with Kinect^{TM}

Beatriz Quintino Ferreira[a], Miguel Griné[a], Duarte Gameiro[a], João Paulo Costeira[a,b] and Beatriz Sousa Santos[c,d]

[a]DEEC, Instituto Superior Técnico, Lisboa, Portugal
[b]Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal
[c]Dep. Electrónica, Telecomunicações e Informática, Universidade de Aveiro, Aveiro, Portugal
[d]Instituto de Engenharia Electrónica e Telemática de Aveiro, Aveiro, Portugal

ABSTRACT

This article presents a solution to volume measurement object packing using 3D cameras (such as the *Microsoft Kinect*^{TM}). We target application scenarios, such as warehouses or distribution and logistics companies, where it is important to promptly compute package volumes, yet high accuracy is not pivotal. Our application automatically detects cuboid objects using the depth camera data and computes their volume and sorting it allowing space optimization. The proposed methodology applies to a point cloud simple computer vision and image processing methods, as connected components, morphological operations and Harris corner detector, producing encouraging results, namely an accuracy in volume measurement of $8mm$. Aspects that can be further improved are identified; nevertheless, the current solution is already promising turning out to be cost effective for the envisaged scenarios.

Keywords: volume measurement, *Microsoft Kinect*^{TM}, 3D depth camera

1. INTRODUCTION

This article presents an approach to volume measurement using *Microsoft Kinect*^{TM}, which has several application scenarios such as warehouses or distribution and logistics companies, where it is important to promptly compute package volumes. For these environments, instead of using the standardized and pricey laser based technology (e.g. laser-range finders which cost several thousands of dollars), we propose a low-cost solution using the widely available plus quite affordable Microsoft *Kinect*^{TM} (with price approximately 100$). In such situations, where the laser high accuracy (around $5mm$) is not mandatory, we can automatically detect boxy objects from the depth camera data and compute their volume, paving the way for further space optimization. The proposed methodology, based on simple computer vision and image processing methods[1] , provides a new and cost-effective solution for the industry. Among the several methods used to process the *Kinect*^{TM} depth camera information, we highlight: RANSAC ("RANdom SAmple Consensus"[2]) which allows to find the planes contained in the scene; connected components[3] in order to segment the object faces; morphological operations[4] to filter the faces and, also, the Harris corner detector[5] to find possible vertices.

The attained results are promising, with accuracy close to the laser based volume measurement commercial systems.

The remainder of the paper is organized as follows. Section 2 includes a brief description of the 3D camera device used (*Kinect*^{TM}), as well as the camera calibration performed. In Section 3 the system work-flow and the methods used to compute the volume of the regular objects are described. The different tests performed and results obtained are presented in Section 4. Lastly, in Section 5 some conclusions are drawn and future work directions are discussed.

Beatriz Quintino Ferreira: beatriz.quintino@tecnico.ulisboa.pt
Miguel Griné: miguel.grine@tecnico.ulisboa.pt
Duarte Gameiro: duarteflgameiro@tecnico.ulisboa.pt

2. *KINECT*^{TM} AND CALIBRATION

Kinect^{TM} is a motion sensing input device by Microsoft®.[6] This device features a RGB camera and depth camera (which comprises an infrared laser projector and a depth sensor) at a very affordable price (approximately, 100$). The recent availability of low-cost Kinect^{TM} sensor provides a valid alternative to other available sensors, namely laser-range finders and stereo-vision systems. The Kinect^{TM} depth sensor brings a fast and relatively high resolution solution for depth sensing, while being considerably cheaper than the aforementioned alternatives. Nevertheless, it has some limitations, specifically, this device was designed to work indoors due to light conditions. Moreover, its range is up to $4m$ and can not acquire valid data at distances smaller than $0.8m$. Figure 1 shows its main components, namely the RGB camera and depth sensors, a multi-array microphone and a motorized tilt.

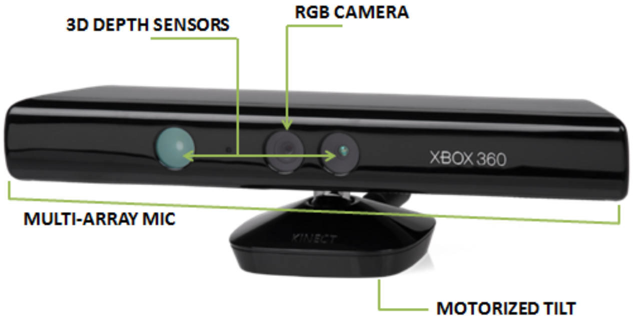

Figure 1. Microsft® Kinect^{TM}: components (adapted from *http://en.wikipedia.org/wiki/File:Xbox-360-Kinect-Standalone.png*)

An important previous step, before using the 3D depth camera data to compute volumes, concerns the camera calibration. The linear camera model which was used to calibrate the Kinect^{TM} camera is presented below (in eq. 1):

$$\begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

where K is the intrinsic parameters matrix, while R and T are the extrinsic parameters (rotational and translational matrices, respectively). The intrinsic parameters are the conversion factors from meters to pixels and the coordinates of the principal point in the new frame. These parameters are denoted as intrinsic parameters of the camera since they only depend on the camera and are independent of its position and orientation.

If we choose a 3D frame that is neither attached to the camera nor located at the optical center of the camera, in other words, with an arbitrary location and orientation, the new coordinates are related to the camera coordinates by a rigid body transformation (a rotation, followed by a translation). Thus extrinsic parameters are R, the rotational matrix, and T, the translational matrix. In our case, since the depth camera referential is the same that the world's referential, we considered that R is equal to identity matrix and T is zero.

The process of calibrating the camera consists, basically, in estimating the camera parameters from an image of the calibration object with known geometry (for instance, a chess board), since it contains a set of easily detected points, which can be used as the calibration points. The camera parameters can be chosen in order to minimize the distance from the points projected by the camera and by the model. Since we have an optimization problem (which is non-convex), we applied a linear regression (least squares), using the pseudo-inverse method to solve it.

We performed the calibration of Kinect^{TM} as described, nevertheless, it is possible to find alternative, ready to use calibrations (e.g.[7] and[8]).

3. VOLUME MEASUREMENT METHODS

To achieve our goal of determining the volume of a cuboid shaped object, it is necessary to measure three edges in order to have the length, width and height of the box. Hence, our approach focuses on finding the inliers of the planes of the scene and, subsequently, the vertices that connect adjacent planes, so that it is possible to determine each edge length. The proposed application was developed in MATLAB® 7.9.0 (R2009b).[9]

Next, the system workflow is presented, in figure 2, illustrating the application high-level operation. The methods are further discussed, afterwards.

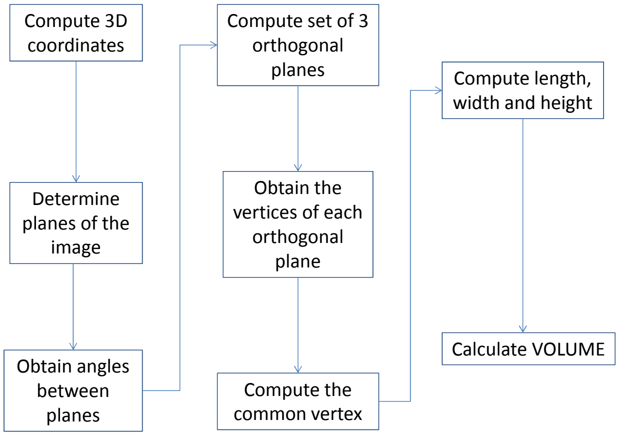
Compute 3D coordinates
Determine planes of the image
Obtain angles between planes
Compute set of 3 orthogonal planes
Obtain the vertices of each orthogonal plane
Compute the common vertex
Compute length, width and height
Calculate VOLUME

Figure 2. Workflow of the developed system, allowing to compute volumes of cuboid objects

Considering a known and clean environment with a cuboid object, for instance a box, both RGB and depth images are acquired from the cameras, as it can be seen in figures 3 and 4 (please note that the position of the box is very important, since the three edges to be computed have to be entirely visible; a solution to this constrain will be further discussed).

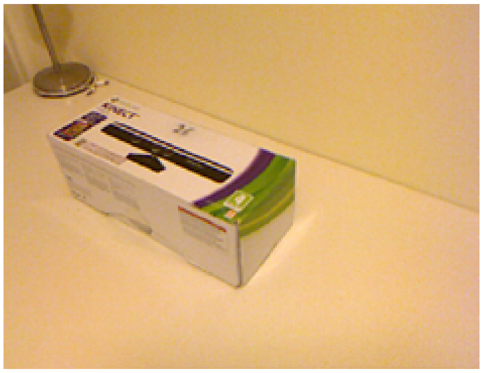

Figure 3. RGB image

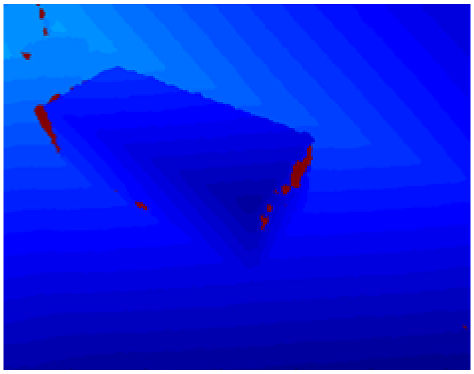

Figure 4. Depth image

One should note that this implies a previous calibration of the *Kinect*^{TM}, using the camera model (as described in Section 2).

From the acquisition step, the depth information is used to compute the 3D coordinates *(X Y Z)*, using again the camera model, and a 3D point cloud representation is obtained. During this process, the invalid point readings caused by points out of range or lighting conditions are removed from the set (these points are depicted in red in figure 4). Invalid readings are retrieved by *Kinect*^{TM} with a special code number, out of the range. With the obtained point cloud, the inliers of the planes shown in figure 5 can be determined applying the RANSAC method.[2]

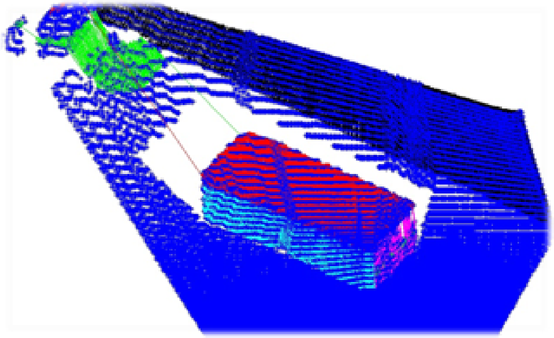

Figure 5. Point cloud with the planes inliers; representing different planes using different colors

Having the inliers of each plane of the scene, considering a clean and known environment, we remove the two planes with more inliers which correspond to the background. Alternatively, in order to segment the object to be measured, we could perform background subtraction. In our first experimental setup (described in Section Results) we verify these assumptions, which in spite of being simplifying assumptions, we believe they are easily applicable in the envisaged scenarios.

Once the planes of the object are delimited, the angle between each pair of neighboring planes is computed. From the obtained angles a set of three adjacent and orthogonal planes is selected so that we can determine the right edges to compute the volume. This selection is a very important step and it is performed using a clique problem approach.[10]

The processing proceeds with the projection of each of the three orthogonal planes delimited into a 2D binary image, which will help the subsequent steps. This projection is performed using labels of connected components.[3] Due to some faults that occur during the capture with the camera, as mentioned in Section 1, morphological operations[4] are applied to filter these 2D images. We used mainly the dilation and closure operations, since the detected faces were irregular and showing some missing data points. This filtering is rather important for the next phase, which consists in applying the Harris corner detector[5] to every segmented face, in order to find possible vertices.

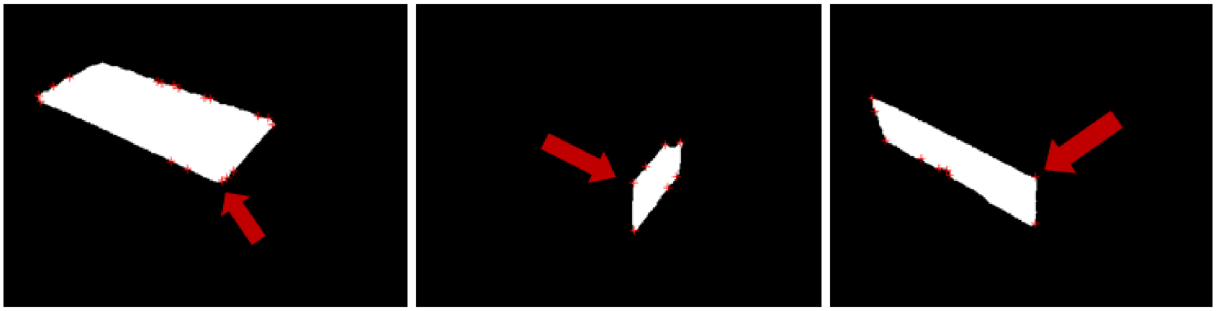

Figure 6. Planes represented as binary images, with the detected corners depicted and the vertex common to the three orthogonal planes pointed out with an arrow

The next step is to compute the corners which are of interest, in order to compute the lengths between them and, then, the final goal: the object volume. To find these special points we used the Harris corner detector; yet, the parameter fine-tuning proved arduous. We tested several values in order to achieve the best sensitivity (find the persued vertices), whereby the final parameters were a combination of a very small threshold with a large radius in non-maximal suppression. After applying the detector, it is crucial to determine which vertex, among all the corners found to be possible vertices, is the common vertex; in other words, we have to look for the vertex which belongs to the three orthogonal planes (emphasized in figure 6). Therefore, we are facing a minimization problem, since we are verifying which vertex has the smaller Euclidean distance to each plane. After that, the next step is to determine the vertex in the other end of each edge computing the maximum Euclidean distance among the corners found for each couple of planes. Then, since we already have the four vertices we need, it is possible to measure the three lengths required to compute the volume of the box, as represented in figure 7. To do that, we just need to map the sets of two vertices (the common vertex of the three orthogonal planes plus the vertex on the opposite side of the edge which are in 2D binary images) into the real world coordinates (3D - point cloud), using the camera model.

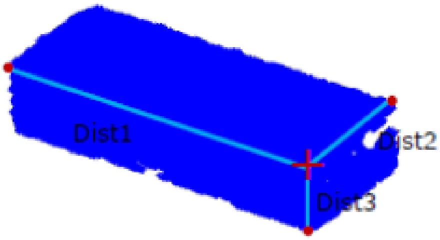

Figure 7. Common vertex and the vertices used to compute the length of the three edges

These lengths are computed through Euclidean distances between each couple of vertices, and the object volume is, thereafter, readily calculated as the product of these lengths.

When the common vertex for the three orthogonal planes is not found, it is not possible to calculate the volume. In such cases the application displays an error message asking the user to move the object so that this vertex can be computed. This allows to improve our solution robustness, since no inconsistent values are obtained as the final result.

4. RESULTS

In order to evaluate the solution, accuracy and precision studies were conducted using data sets comprising several cuboid objects. The experimental setup used included a $Kinect^{TM}$ placed at about $1m$ distance from the object at a level slightly above the box, so that it is easy to obtain an adequate view of the three main faces of the object. This was intended to replicate the positioning in an real application scenario.
Our data set comprised four boxes of different sizes and proportions, however, for all cases, the largest edge was smaller than $1m$.

Figures from 8 to 10 show the histograms corresponding to the three edges of a specific box, for which we performed 20 tests. The box real dimensions were $0.375m$, $0.145m$ and $0.118m$, length, width and height, respectively. As the histograms show, the maximum error is 0.008m for the length (2%), $0.007m$ for width (4.8%) and 0,018m for height (15%). These results were found to be representative considering all the performed tests.

Figure 11 shows the volume computed, revealing that the maximum error is $0.0010m^3$ (15,6%), as the real volume was $0.00642m^3$.

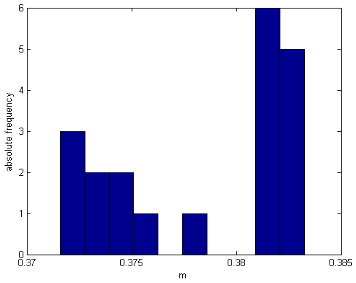

Figure 8. Histogram of 20 measurements of a box length

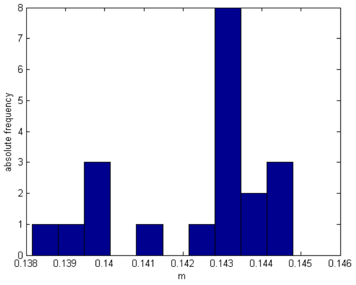

Figure 9. Histogram of 20 measurements of a box width

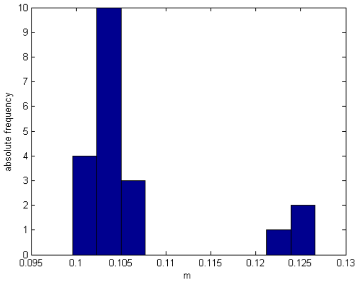

Figure 10. Histogram of 20 measurements of a box height

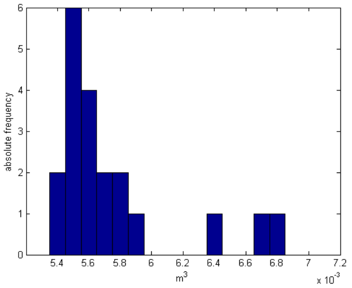

Figure 11. Histogram of 20 computation of a box volume

5. CONCLUSIONS AND FUTURE WORK

This article presents a methodology to compute volumes of cuboid objects based on the data acquired by the depth camera of *Microsoft Kinect*^{TM}. The volumes of the objects from our data set were successfully computed with an accuracy suggesting an adequate compromise between performance and cost regarding the identified application scenarios.

Based on the experiments performed, we believe that better accuracy and precision could be obtained acting upon the following processing steps: improving the Harris corner detector parameters and exploring RGB information. Specifically, the RGB camera data could be used to help segmenting the objects (as faces from the same object tend to have a similar color palette distinct from the background), also allowing the segmentation of superimposed objects, as long as they have different colors. Another line of work would be improving the independence from the background, so that our solution would have a good performance in any environment, regardless the background (thus eliminating the restriction of a known and clean environment). Likewise, background subtraction could be applied using the RGB information.

Moreover, a second experimental setup using an array of two or more depth cameras is envisaged. Provided that all cameras are calibrated and the transformation among them is known, this setup could produce more precise results and enable the usage in additional scenarios, since with more cameras a wider region is covered, thus allowing measuring larger volumes.

REFERENCES

1. R. Szeliski, *Computer Vision: Algorithms and Applications*, draft Springer, 2009.
2. M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Comm. of the ACM* **24**(6), pp. 381–395, 1981.
3. H. Samet and M. Tamminen, “Efficient component labeling of images of arbitrary dimension represented by linear bintrees,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **10**(4), pp. 579–586, 1988.
4. J. Serra and P. Soille, “Mathematical Morphology and Its Applications to Image Processing,” in *Proceedings of the 2nd International Symposium on Mathematical Morphology (ISMM’94)*, 1994.
5. C. Harris and M. Stephens, “A combined corner and edge detector,” in *Proceedings of the 4th Alvey Vision Conference*, pp. 147–151, 1988.
6. “Kinect sensor.” `http://msdn.microsoft.com/en-us/library/hh438998.aspx`. Accessed: November 2012.
7. “Kinect Calibration Toolbox.” `http://www.ee.oulu.fi/~dherrera/kinect/`. Accessed: November 2013.
8. “Kinect Calibration.” `http://www.mrpt.org/tutorials/programming/miscellaneous/kinect-calibration/`. Accessed: November 2012.
9. “MATLAB - The Language of Technical Computing.” `http://www.mathworks.com/products/matlab/`. Accessed: December 2013.
10. R. E. Tarjan and A. E. Trojanowski, “Finding a maximum independent set,” *SIAM Journal on Computing* **6**(3), pp. 537–546, 1977.