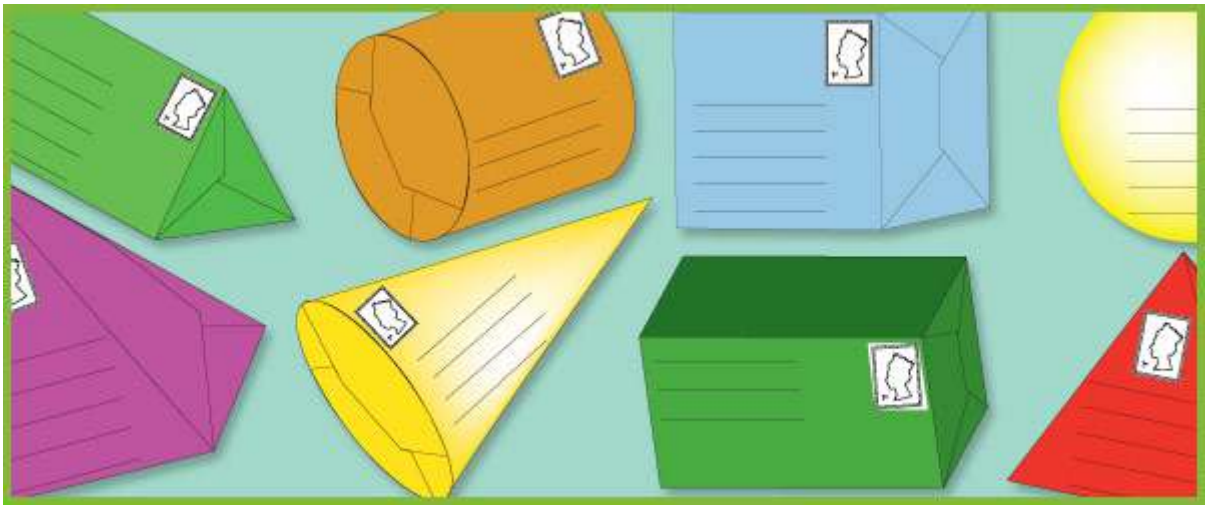


https://www.earlylearninghq.org.uk/role-play-2/post-office-themes/3d-shape-parcels-letter-templates/

PARCEL CLASSIFICATION & DIMENSIONING

Abhijeet Bhatikar

Daphne Tsatsoulis

Nils Heyman-Dewitte

William Diggin

31 October 2020

THE OPPORTUNITY

Airline cargo companies only sell 85%-90% of their maximum airplane cargo capacity due to the inability to accurately measure their cargo. This problem has been identified in the wider global cargo industry.

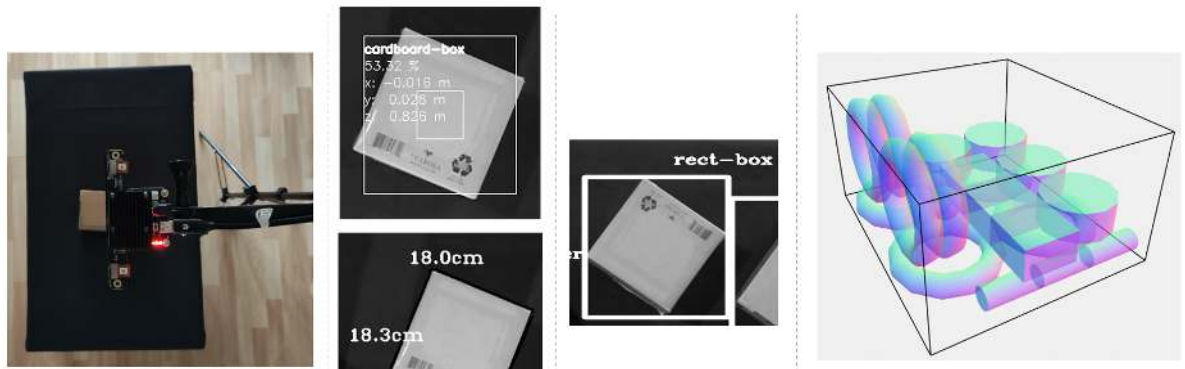
Total global revenues of cargo airlines average around USD 103 billion per year over the last 3 years, the highest it's been since 2004 [**Error! Reference source not found.**]. The ability to efficiently fill cargo airplanes to the maximum can result in a lower carbon footprint and higher profit margins per tonne of shipped airplane cargo.

The reason they still miss out on 10% to 15% airplane cargo filling efficiency is two-fold:

1. Many cargo companies do not measure their cargo accurately at a large scale using the newest technology. They just describe their cargo with width, height and depth at best. In many cases, a manual measuring tape is used to make these guesstimates.
2. Even when they do measure their cargo on a more granular level, they mostly always fit cuboids on top of irregularly shaped objects. For example, a cylindrical barrel's dimensions are up to the nearest 3D cube that fits this barrel (width x length x height).

After this, even if they do use efficient heuristic-based loading optimizers, the damage is already done. All the cargo has been rounded up to the nearest cuboid, so automatically leaving a significant amount of small spaces between stacked (irregular) cargo. This makes us conclude there is a massive opportunity to more efficiently dimension cargo at a granular level - filling up the little (or massive) gaps in the freight cargo industry.

OUR SOLUTION



Step 1: Capture frames using the DepthAI USB3

Step 2: Detect cardboard parcel and compute width, length & height

Step 3: Identify the shape of the parcel

Step 4: Find the optimal container layout for all measured parcels

Figure 1: Our solution for finding the optimal container layout leverages the DepthAI USB3 camera (OAK-D) and the MobileNet SSD architecture. For each package we first capture frames using the DepthAI camera in the setup shown on the left. We then detect the parcel using a custom model trained based on the MobileNet SSD, determine the shape, calculate the width, length and height. Once all packages have been measured and the shape identified, we can compute the optimal container packing layout using EB-AFIT algorithm.

We have built an end-to-end proof of concept for measuring and loading packages into cargo containers. Our solution leverages the DepthAI USB3 (OAK-D) [12] camera to accurately determine the shape, then measures the width, length and height of packages for cargo shipment. We have also built a software tool that finds the optimal arrangement of the batch of packages given the container's size. As shown in *Figure 1*, once a parcel is put in the field of view, the trained AI model detects the parcel e.g. cardboard box. The application feeds the

frame from the right stereo camera to our parcel dimension calculator and our shape detector modules as shown in *Figure 2*. The parcel dimensions and shapes are aggregated and fed into a container stacking application to compute and visualize the most efficient package layout. We detail our method below.

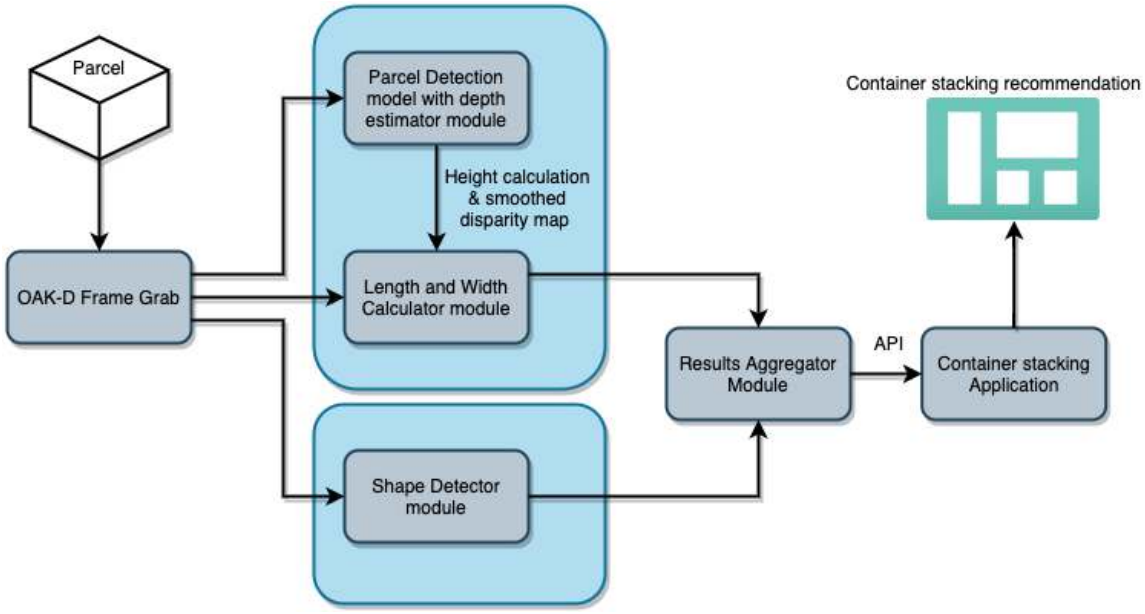


Figure 2: High-level architecture of the proposed solution. The input is a captured DepthAI USB3 (OAK-D) frame and the output is a visual container stacking recommendation.

PARCEL DETECTION MODEL & HEIGHT CALCULATION

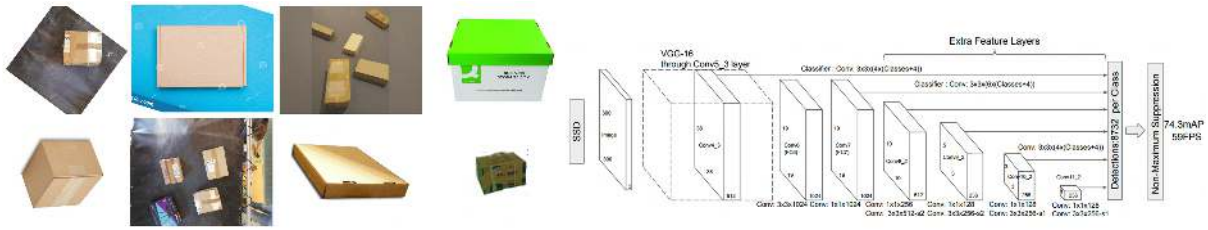


Figure 3: We trained our parcel detector using the MobileNet SSD architecture ([Error! Reference source not found.], [7]. [Error! Reference source not found.]) on the cardboard box training set we collected (examples seen above). This dataset can be replaced with other training data to expand our solution's performance to new categories.

The parcel detector model we trained uses the *MobileNet SSD architecture* ([**Error! Reference source not found.**], [7], [11]) and is trained [14] on a custom dataset of cardboard boxes, examples can be seen in *Figure 3*. The dataset [13] has 305 images all labelled in the Pascal VOC format. This dataset currently only has cardboard boxes but could easily be expanded to include other items e.g. glass boxes, musical instruments cases, etc.

The AI model predictions were set on a confidence threshold of 0.5 and the bounding boxes that were extracted are used to compute the depth of the parcel. *Figure 4* shows the output of the parcel detector with the depth channel output in Image A. We use the DepthAI API to get the average depth within the neural inference bounding box. We adjust the disparity confidence to get steady readings within the bounding box detected. We then calculate the height of the parcel.

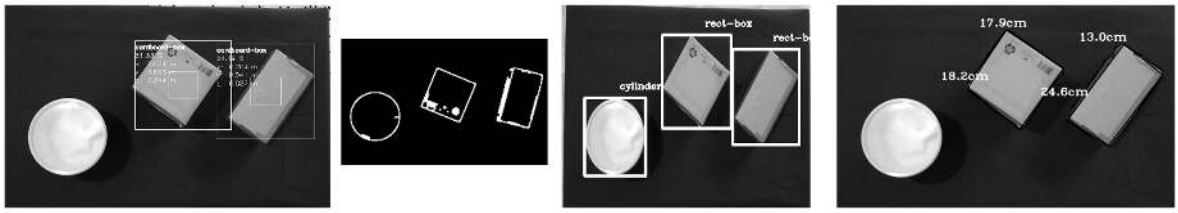


Image A *Image B* *Image C* *Image D*

Figure 4: Once parcels were detected (Image A) we used a Canny edge detector to identify the shape of the parcel (Image B) – for example, a parcel with four points is a box, a parcel with 8 points is a cylinder (Image C). Lastly, we report the width, length and height of the parcel (Image D).

WIDTH & LENGTH CALCULATION

The right stereo camera frame is also passed to our width and length calculation module. This module consists of 3 main stages: ***contour extraction, warping and dimension computation***. A canny edge detector is first used to determine the edges in the image (*Figure 4, Image B*). This result is then used to determine all the closed contours in the image using the OpenCV method of finding Contours [**Error! Reference source not found.**], [5].

We identify the largest closed contour which represents the supporting surface and consider smaller closed contours to be the parcels. To correct for any skew in the image we warp the image based upon the known dimensions and perspective of the surface [**Error! Reference source not found.**]. We are then able to extract the width and length of the parcels by calculating the distance between the four points of their contours. We display the computed dimensions in the image as shown in *Figure 4*, Image D.

The three dimensions (width, length and height) of the parcel are then passed to a result aggregation module that serializes the data for the 3D container stacking software API.

SHAPE DETECTION

The shape detector module determines the shape of the parcel, a main input to the 3D stacking software. It also uses the edge detection and contour approximation method described above. These contours are used to determine if the parcel is a cuboid or cylinder by counting the points of the contour. If the object has 4 points it is a cuboid and if it has 8 points it is a cylinder. Example results can be seen in *Figure 4, Image C*. The shape for each parcel is then passed to the result aggregation module.

3D CONTAINER STACKING SOFTWARE

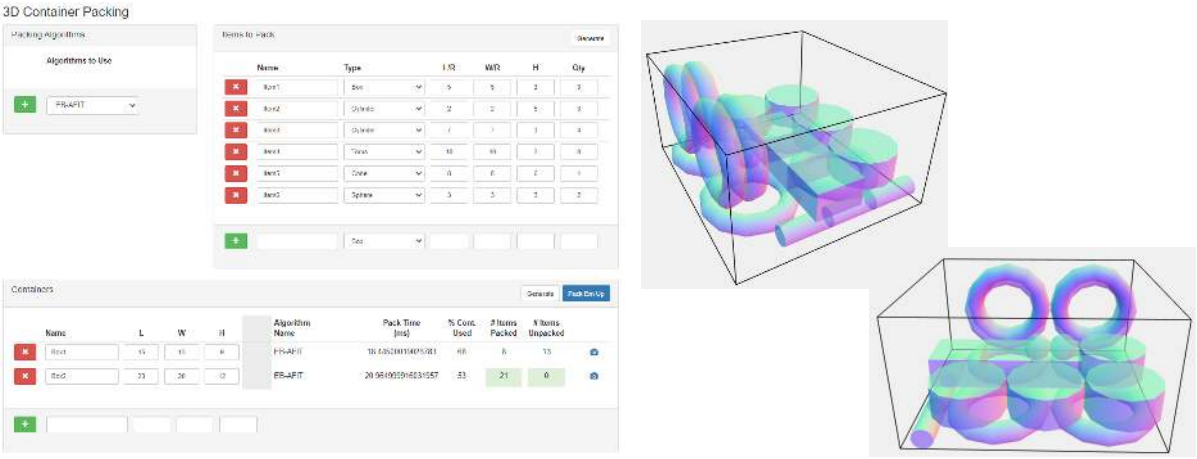


Figure 5: Once all packages have been measured using the DepthAI USB3 we use the EB-AFIT packing algorithm to find the optimal layout in the container. The images here show an example of the result given several packages of varying shape.

The parcel shape and dimensions are transferred to a container stacking software (forked from [**Error! Reference source not found.**]). The approach taken for the packaging is based on [**Error! Reference source not found.**]. The 3D container stacking software (seen in *Figure 5*) takes the parcel identifier, shape and dimensions as input along with the dimensions of the container into which the parcels would be loaded. It uses the EB-AFIT packing algorithm to find the optimal layout for the container. The final output is a visualisation of how the parcels should be stacked inside the container in order to optimally use the provided space. The software also reports the percentage of the container used and the number of items it was able to pack into the container, as it is possible that they will not all fit.

RESULTS

Parcel specs	Ground truth dimensions Parcel 1	Ground truth dimensions Parcel 2	Calculated dimensions Parcel 1	Calculated dimensions Parcel 2	Average error	Average error %
Width	*12.1cm*	*14.7cm*	*13.4cm*	*17.7cm*	*2.2cm*	*~16%*
Length	*22.0cm*	*15.6cm*	*24.5cm*	*18.3cm*	*2.6cm*	*~14%*
Height	*6 cm*	*15 cm*	*7 cm*	*9-18 cm*	*2-3.5cm*	*~19%-33%*

We tested our system on two cardboard boxes and a cylindrical container as shown in *Figure 4*. The camera was placed overhead a table and we used both natural and artificial lighting sources based on the time of day. A visualization of our setup is in *Figure 1.*

The ground truth and predicted parcel dimensions can be seen in the table above. Generally, our system did well with an average error of about 2.5cm across the three dimensions. The height calculation was the most challenging because of issues with lighting. When we had reliably strong natural lighting the results were most accurate, however our indoor lighting was not sufficient to get the best results.

Generally, these results are quite promising, and we are pleased with the accuracy of the measurement we were able to extract using the camera. The main learning from these experiments was that the lighting greatly influences the accuracy of our results.

LIMITATIONS & FUTURE WORK

We have designed and created a solution to automatically calculate the 3D size of rectangular packages and optimally place them in a container. The process we have designed can be expanded and improved to create a more robust and generalizable solution.

PARCEL PREDICTION AND DIMENSIONING

Our current solution has been designed end-to-end for rectangular packages. To enable other shapes of packages we would expand the training data we used for our custom MobileNet SSD parcel detection model to include additional items and extend the shape detector beyond cuboid and cylindrical shapes.

We could also modify the way we calculate the dimensions of the parcel by calculating the surface area of the detected parcel instead of the specific dimensions. This could help mitigate any errors that arise due to the physical set up or camera angle.

Additionally, we could use the weighted least squares (WLS) filter that is a well-known edge preserving smoothing technique, but its weights highly depend on the image gradients. It helps to calculate the smoothing weights for pixels based on both their isotropy and gradients. The WLS filter would smooth things out to get a steady disparity map and improved measurements on depth.

PHYSICAL SETUP

One of the dominant challenges we faced was the limitation of our physical set up. To improve our results going forward we would need to adjust the set up so that there was more uniform lighting and less background noise. This data would enhance our calculations. We could also use multiple viewpoints of a package to improve results by either leveraging multiple cameras or a rotating surface for the parcel.

CONTAINER PACKING

Our current packing algorithm is a CAD based packing solution. This could be replaced with a volumetric morphological-based packing algorithm. This may require more than one depth camera or a turntable assembly to generate the appropriate surface and voxel dataset.

FUTURE STEPS

The last step in bringing our solution to life would be to orchestrate the filling of a container with the packages in the right layout. Since our container packing solution works on a batch of package measurements every package would have to be uniquely recognizable and put in a sort of queue for loading. To uniquely identify a package, we could use RFID tags or a unique digital fingerprint extracted by taking an image of the package as offered by Alitheon [8].

REFERENCES

1. https://www.statista.com/statistics/564658/worldwide-revenue-of-air-cargo-traffic/

2. https://github.com/wild-ig/packman

3. https://www.researchgate.net/publication/262832429_The_distributor%27s_three-dimensional_pallet-packing_problem_A_human_intelligence-based_heuristic_approach

   Baltacioglu, Ethan; Moore, James T.; Hill, Raymond R.; The distributor's three-dimensional pallet-packing problem: A human intelligence-based heuristic approach; International Journal of Operational Research, March 2006;

4. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html

5. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html

6. https://github.com/IntelAI/models/tree/master/models/object_detection/tensorflow/ssd-mobilenet/inference

7. https://github.com/chuanqi305/MobileNet-SSD

8. https://www.alitheon.com/featureprint-platform

9. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_geometric_transformations/py_geometric_transformations.html

10. https://github.com/davidmchapman/3DContainerPacking

11. https://wzding.github.io/wzding.github.io/projects/Object_Detection_MobileNets_SSD.html

12. https://docs.luxonis.com/products/bw1098obc

13. https://jfids.jasaplus.com/repo/cardboard.tar.bz2

14. https://colab.research.google.com/github/luxonis/depthai-ml-training/blob/master/colab-notebooks/Easy_Object_Detection_With_Custom_Data_Demo_Training.ipynb