

# Volume estimation via integrating on a curve fitted point cloud

**HiPEDS 2018 Cohort:** G. Bisbas, L. Castiglione, D. Grumberg, S. Karolčík, L. Keeble, D. Kulon, B. Kwan, C. McMeel, R. Miles, J. Ortiz, N. Perez-Nieves, V. Pham Ngoc, J. Vandebon, D. Vink

Imperial College London



**Imperial College  
London**

October 29, 2018



## HiPEDS Group workflow

- Cohort meetings on a regular basis
- Identify our goals and split into subgroups
- Integrate our progress
- Redefine goals

## Point cloud integration team overall checkpoints

- Capture images
- Extract point cloud
- Fit a curve
- Find the volume inside

*More details in the next slides...*

# The problem and the goal



Figure: Royal vans. *Courtesy of <https://www.pressandjournal.co.uk>*

- **Problem:** Packaging in vans is not optimal → lots of empty space
- **Goal:** Fast estimation of available volume to ensure optimal packaging

# The hardware - Choosing the camera fitting our needs best

Three (3) cameras were considered. Pros and cons of each were weighted and the group ended up choosing Intel RealSense D435.



Intel Realsense SR300 is better for indoor locations due to infrared light while Zed Stereo Camera is better for outdoors. Our experiments using both of them, reassured this notion. After excluding the camera that didn't work, we settled on the Realsense D435 due to its superior price point and lack of dependence on environmental factors.

# Extracting the point cloud

Luca here you go : Add photos and something about your code

- 
- 
-

# Denoising the point cloud

Extracted point clouds are often very noisy, so, a need for removing outliers is needed.

- Step 1: Hard-cut denoising of points that are very far away of the mass centre.
- Step 2: Use a nearest neighbours approach with a predefined threshold to remove more outliers.

Resulted point clouds were found to be **95%** correctly denoised via simple parameter tuning.

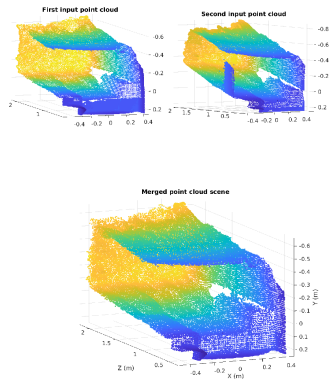
# The ICP algorithm

The Iterative Closest Points algorithm was used to merge the different point clouds extracted from the camera.

For each point in the source point cloud, match the closest point in the reference point cloud (or a selected set).

- 1 Estimate the combination of rotation and translation using a root mean square point to point distance metric minimization technique which will best align each source point to its match found in the previous step. This step may also involve weighting points and rejecting outliers prior to alignment.
- 2 Transform the source points using the obtained transformation.
- 3 Iterate (re-associate the points, and so on).

# Merging point clouds



**Merging procedure:** 2 steps

1st step: 4 to 2

2nd step: 2 to 1

**Figure:** Intermediate merged pointclouds (left) and final point cloud (right).  
Denoised and downsampled version of final point cloud (bottom).



# Curve fitting with Linear Interpolation

Using MATLAB's Curve Fitting Toolbox, we managed to

- Fit a polynomial surface
- Step 2: Use a nearest neighbours approach with a predefined threshold to remove more outliers.

Resulted point clouds were found to be **95%** correctly denoised via simple parameter tuning.

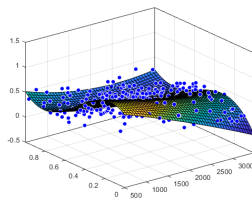


Figure: A simple curve fitting

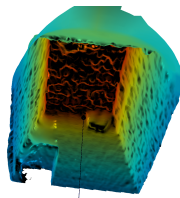


Figure: A full fitted surface

# Integration - Keep it simple scientist

$$\iiint_V f(x, y, z) \, dx \, dy \, dz \quad (1)$$

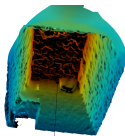


Figure: A full fitted surface

$$\iiint_V f(x, y, z) \, dx \, dy \, dz \quad (2)$$



Figure: Pre-Calculus of Archimedes

# Results

# References

# Acknowledgements

This project was proposed and supported by Royal Mail, EPSRC and Imperial College London. A special thanks to Jeremy Bradley and Ben Glocker for their support and advice throughout.