Detecting FEM geometry using Machine Learning

Master Thesis Defense, Nick Scheider, 26.03.2020

TECHNISCHE UNIVERSITÄT DRESDEN

Dresden Database Systems Group

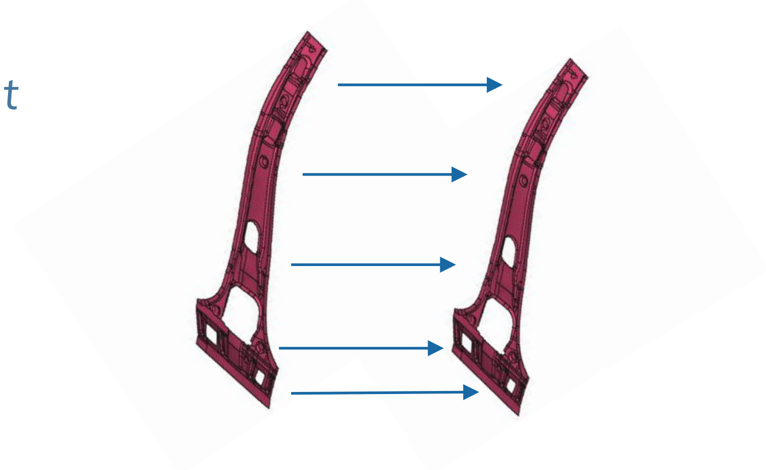
SCALE IT-Solutions for CAE

Motivation

Dresden Database Systems Group

Supporting engineers in the CAE development process

Design decisions based on historical data



Machine learning algorithms

CAD
- Creation of parts
- Generate areas

Pre-processing
- Defining the model
- Discretization
- Environmental factors

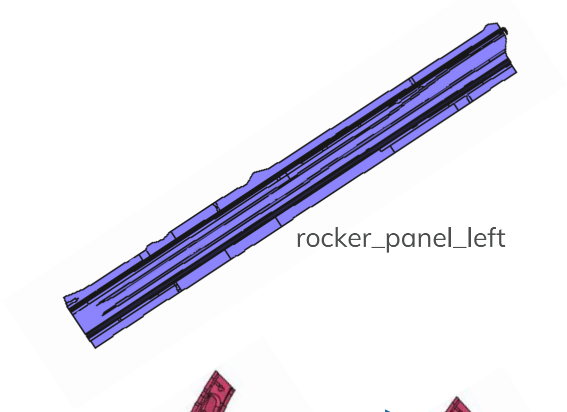
Solver
- Solving equation system (simulation)

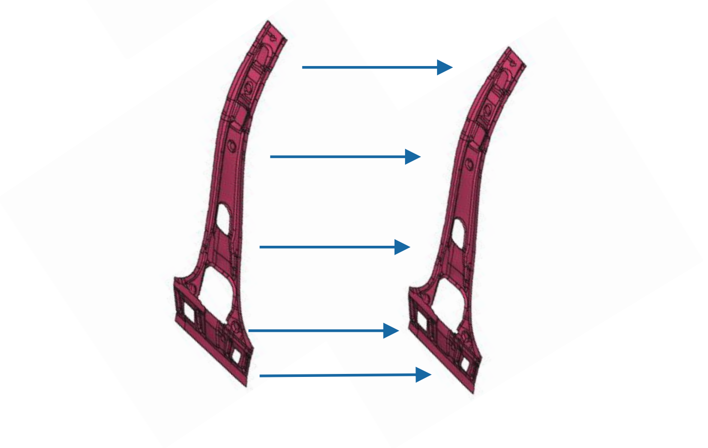
Post-processing
- Visualization
- Analysis

Use cases

Use Case

- Automatic labeling system
- Automatic sorting of part lists
- Recommendation system for spot weld parameter
- Recommendation system of better part constructions
- Segmentation of part groups





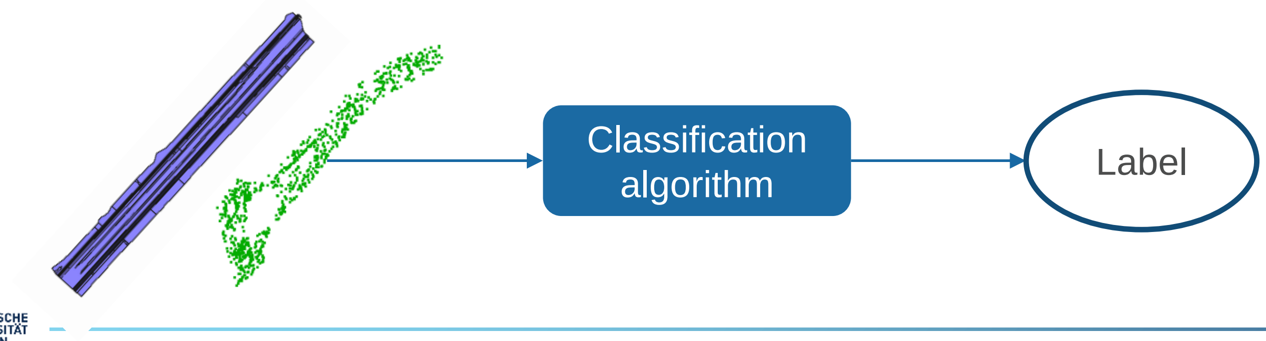
Tasks

Research on approaches in the field of 3D geometric data classification

Conception of a FEM data pre-processing pipeline

Evaluation of the approaches based on the extracted data

Prototypical implementation of a use case



FEM data structure

FEM data structure

Dresden Database Systems Group

Two types used:

- LS-Dyna
- PAM-Crash

Consists of keywords and data blocks

→ database structure

Geometric data under the keyword

**NODE*

```
*PART
aluminum block
$-------+---------+---------+---------+---------+---------+---------+---------+
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1
*SECTION_SOLID
$-------+---------+---------+---------+---------+---------+---------+---------+
$    SECID    ELFORM       AET
         1
*MAT_ELASTIC
$-------+---------+---------+---------+---------+---------+---------+---------+
$      MID        RO         E        PR        DA        DB         K
         1     2700.   70.e+09        .3
*ELEMENT_SOLID
$-----+-------+-------+-------+-------+-------+-------+-------+-------+-------+
$    EID     PID      N1      N2      N3      N4      N5      N6      N7      N8
       1       1       1       2       3       4       5       6       7       8
*NODE
$-----+---------------+---------------+---------------+-------+-------+-------+
$    NID               X               Y               Z      TC      RC
       1              0.              0.              0.       7       7
       2              1.              0.              0.       5       0
       3              1.              1.              0.       3       0
       4              0.              1.              0.       6       0
       5              0.              0.              1.       4       0
       6              1.              0.              1.       2       0
       7              1.              1.              1.       0       0
       8              0.              1.              1.       1       0
```

[1]

FEM data structure

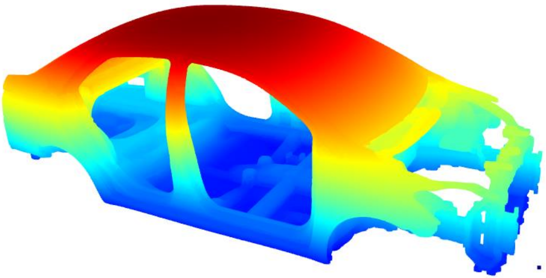
Dresden Database Systems Group

Goal

- Extract 3D geometric data for every part (point cloud)

→ Extract all nodes from all elements of a part

Collect data from more than one car



```
*PART
aluminum block
$--------+---------+---------+---------+---------+---------+---------+---------+
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1
*SECTION_SOLID
$--------+---------+---------+---------+---------+---------+---------+---------+
$    SECID    ELFORM       AET
         1
*MAT_ELASTIC
$--------+---------+---------+---------+---------+---------+---------+---------+
$      MID        RO         E        PR        DA        DB         K
         1     2700.   70.e+09        .3
*ELEMENT_SOLID
$------+-------+-------+-------+-------+-------+-------+-------+-------+-------+
$    EID     PID      N1      N2      N3      N4      N5      N6      N7      N8
       1       1       1       2       3       4       5       6       7       8
*NODE
$------+---------------+---------------+---------------+-------+-------+
$    NID               X               Y               Z      TC      RC
       1              0.              0.              0.       7       7
       2              1.              0.              0.       5       0
       3              1.              1.              0.       3       0
       4              0.              1.              0.       6       0
       5              0.              0.              1.       4       0
       6              1.              0.              1.       2       0
       7              1.              1.              1.       0       0
       8              0.              1.              1.       1       0
```

[1]

Data pre-processing

Data selection

Dresden Database Systems Group

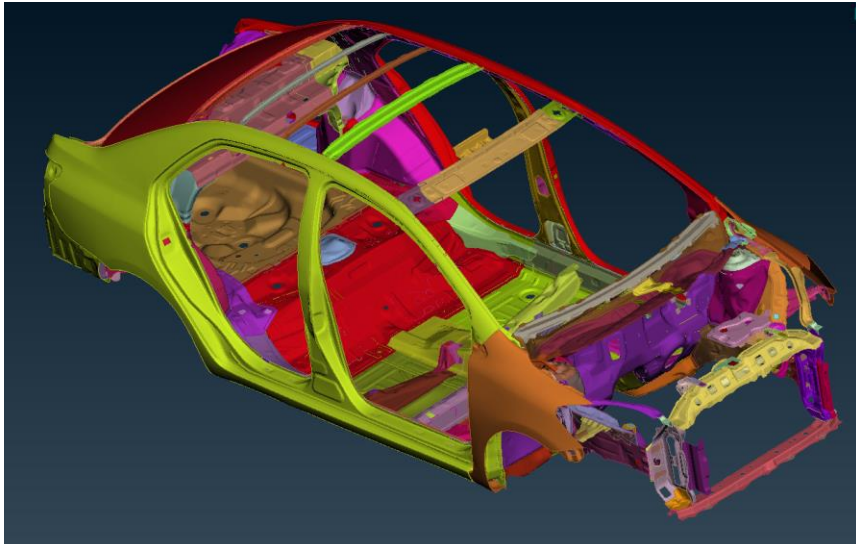
A lot of FEM files extracted from LoCo

- Few differences between files of the same car

Take only one FEM file per car model

→ Six different car models

- Five Audi models and one Toyota Yaris

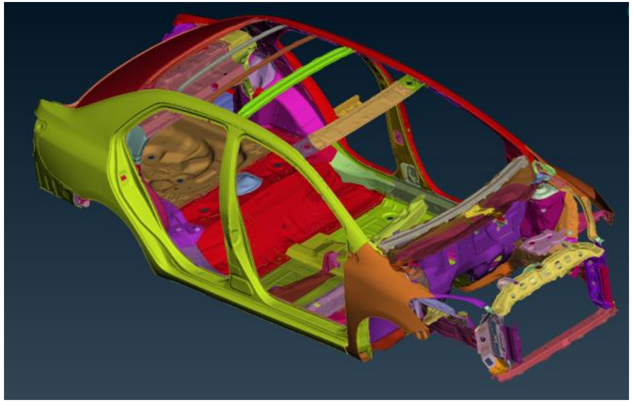


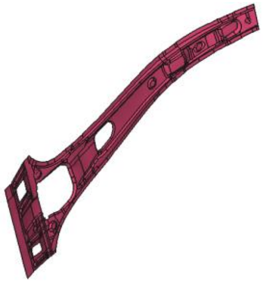
Parsing

Extract 3D data of parts of a car model

Challenges

- Models contain only a subset of the same parts
- No uniform naming of car parts
- Small amount of car models / samples



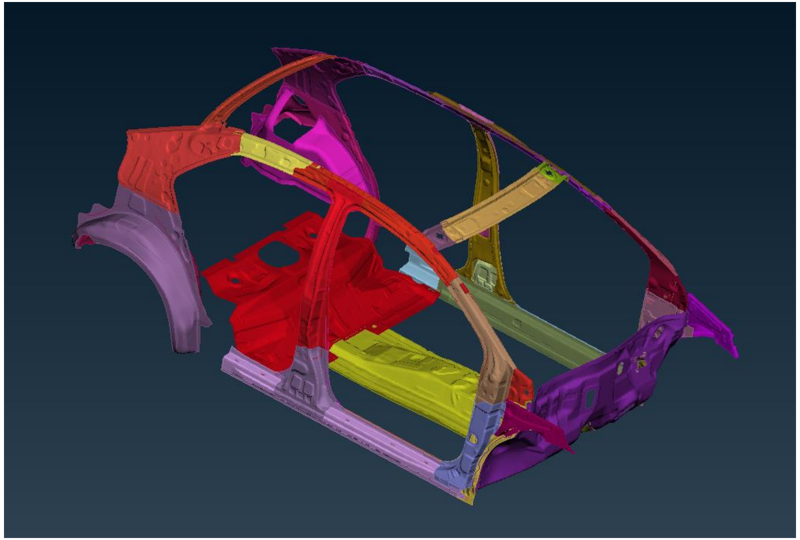


Challenges

Dresden Database Systems Group

Challenges

- **Models contain only a subset of the same parts**
- **No uniform naming of car parts**
- Small amount of car models / samples



Extract only a subset of parts

- Human extracted subset
- Human labeled car parts
  - bpillar_inner_left

Sampling

Challenges:

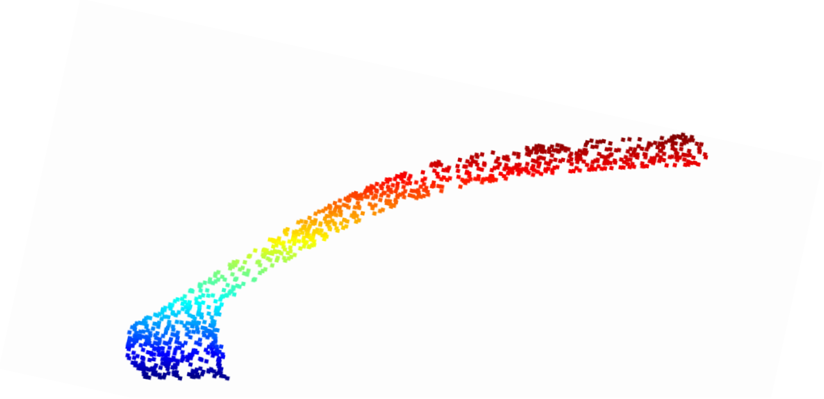
- **Car parts consists of a different number of points**
- **Small amount of car models / samples**

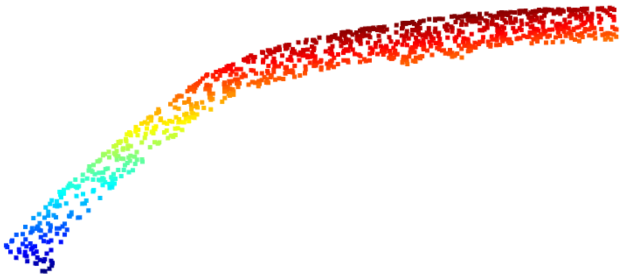
Uniform sampling of the part surface

- Using barycentric coordinates
- Latin Hypercube Sampling

→ *Point clouds with fixed point number*

→ *Generate more than one sample per part*





Normalization

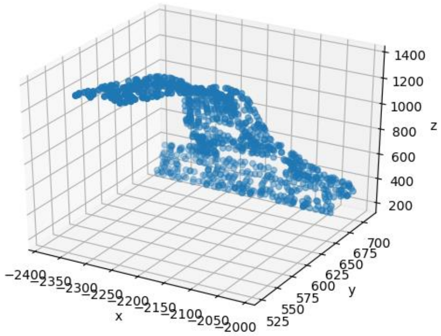
Normalize point clouds

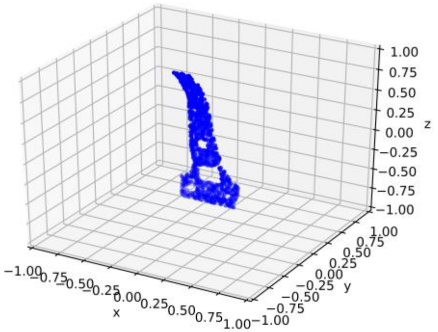
- Smaller values → faster processing
- Standard for 3D geometric data

Usage of mean normalization

- Calculate centroid of point cloud
- Substract centroid → move pc to origin
- Calculate max distance / divide by max distance

$$p\prime = \frac{p - \mu_p}{dist_{max}}$$





Pre-processing pipeline

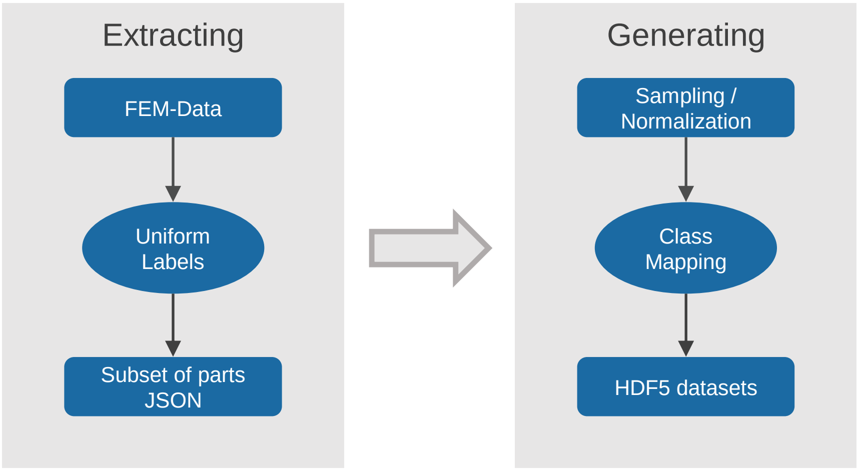
Dresden Database Systems Group

Extraction step

- Parse every FEM model
- Extract subset of car parts and safe as JSON file

Generation step

- Sample and normalize every part in folder
- Mapping of part and corresponding class
- Safe as HDF5 dataset



Datasets

Dresden Database Systems Group

6 HDF5 datasets one per model

More than one sample per part

5 datasets used as trainings data

- 10240 samples

Audi FM3 used as test dataset

- 1024 samples (10 %)

Number of classes depends on labeling

dataset	number of parts	Number of classes	number of samples
Audi FM1	31	(15, 16, 13)	2.048
Audi FM2	31	(15, 16, 13)	2.048
Audi FM3	35	(15, 16, 13)	1.024
Audi FM4	33	(15, 16, 13)	2.048
Audi FM5	23	(15, 16, 13)	2.048
Toyota Yaris	29	(15, 16, 13)	2.048

Deep learning architectures

PointNet

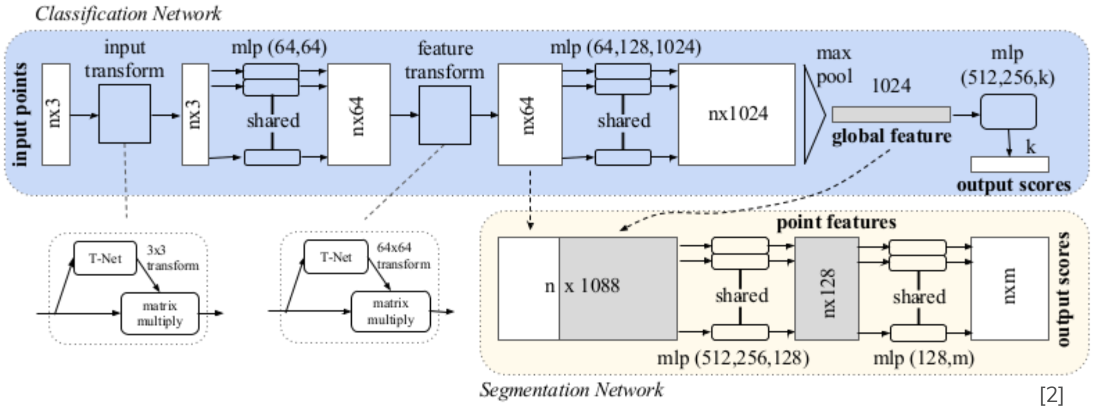
Dresden Database Systems Group

DL architecture for point cloud classification and segmentation

- Charles R. Qi et al. 2016
- Input: point cloud as nx3 array (n point with x, y, z coordinates)
- Output: k class scores
- Invariant to point order
- Invariant to rotation and translation

3.5M parameters

Linear complexity to number of input points



[2]

3D modified Fisher Vectors

DL architecture with new point cloud representation

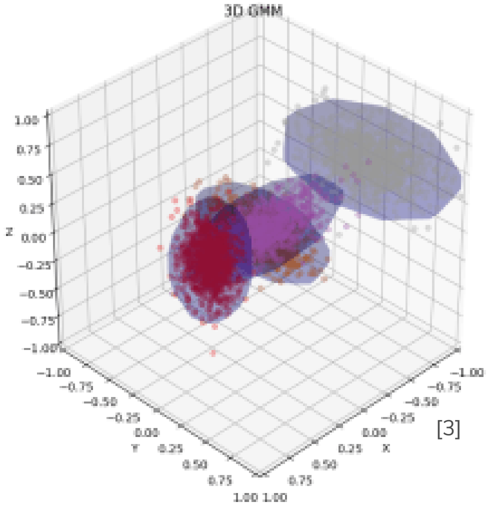
- Itzik Ben-Shabat 2018
- Based on GMM and Fisher Vectors

Gaussian Mixture Model

- Probability distribution of several Gaussians

Fisher Vector

- Describe points by deviation from GMM



[3]

3D modified Fisher Vectors

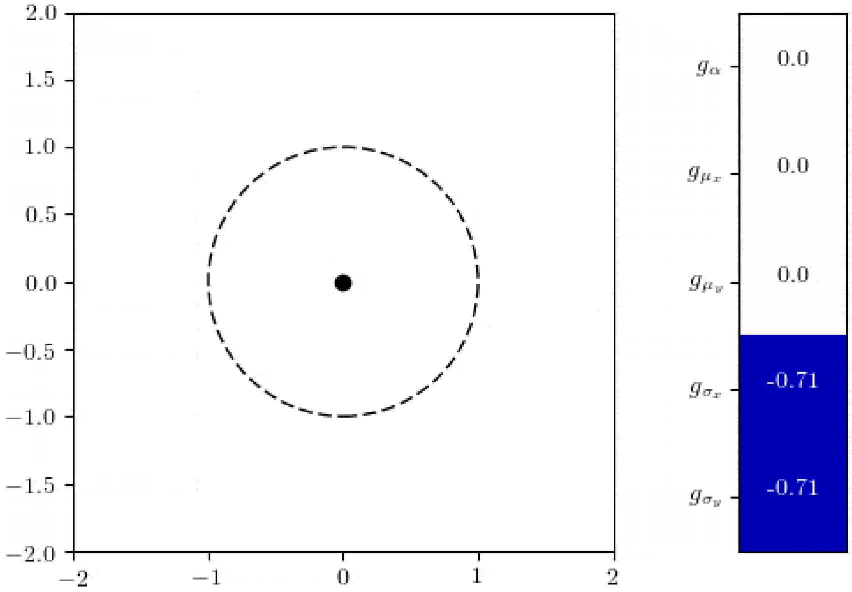
Fisher Vector components

- Normalized gradients w.r.t. Gaussian parameters

$$\mathcal{G}_{\alpha_k}^X = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} (\gamma_t(k) - w_k)$$

$$\mathcal{G}_{\mu_k}^X = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) (\frac{p_t - \mu_k}{\sigma_k})$$

$$\mathcal{G}_{\sigma_k}^X = \frac{1}{\sqrt{2w_k}} \sum_{t=1}^{T} \gamma_t(k) [\frac{(p_t - \mu_k)^2}{\sigma_k^2} - 1]$$



[4]

3D modified Fisher Vectors

Dresden Database Systems Group

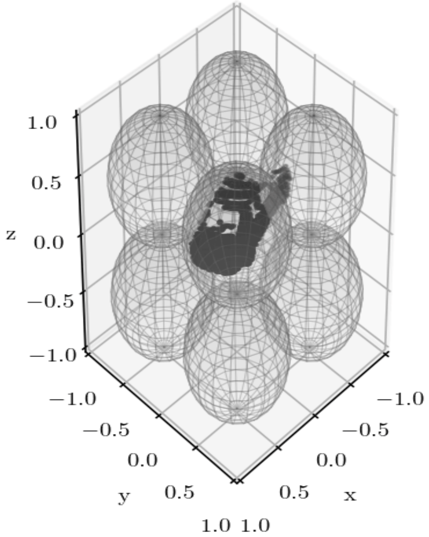
Use GMM on a grid with fixed means and weights

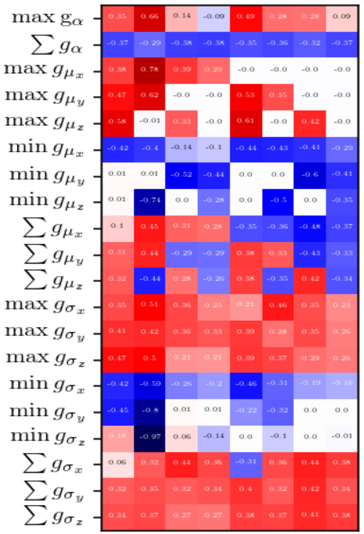
→ Representation of [-1, 1] unit sphere

Fisher Vector for every Gaussian

→ Fisher Matrix

→ New form to represent 3D point clouds





[5]

$$3DmFV_\lambda^X = \begin{bmatrix} \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(p_t)\Big|_{\lambda=\alpha,\mu,\sigma} \\ \max_t(L_\lambda \nabla_\lambda \log u_\lambda(p_t)|_{\lambda=\alpha,\mu,\sigma} \\ \min_t(L_\lambda \nabla_\lambda \log u_\lambda(p_t))|_{\lambda=\mu,\sigma} \end{bmatrix}$$

3DmFV architecture

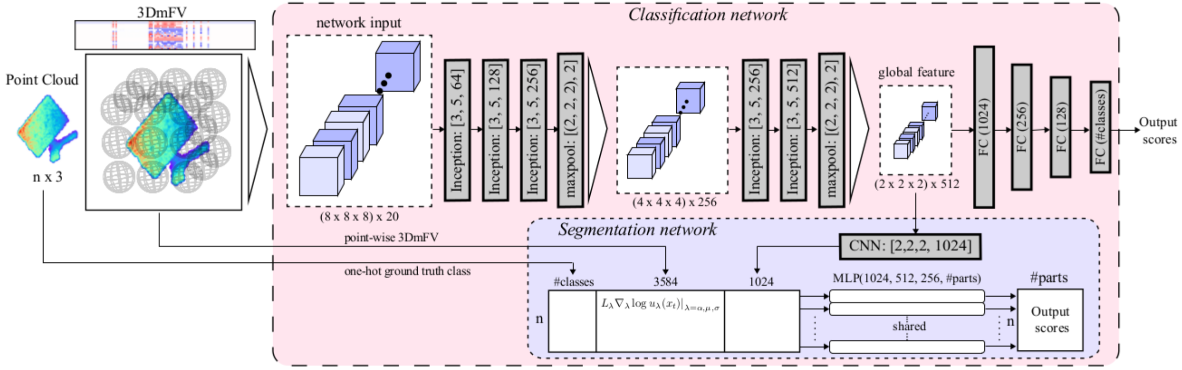
DL architecture with 3D CNN

- Input: nx3 point cloud (transformed into 3DmFV representation)
- Output: k class scores
- Invariant to point order
- Invariant to rotation and translation

Usage of inception networks

- CNN architecture with different filter size

4.6M parameters



[6]

Evaluation

PointNet vs. 3DmFV

Dresden Database Systems Group

Setup

- Hyperparamter nearly identical
- Trained on the same computer for 100 epochs
- After 1 epoch
  → Test with Audi FM3 dataset

Three different Benchmarks

- Coarse part groups
- Distinction between left and right parts
- Distinction between inner and outer parts

→ To analyze the limits of the approaches

	PointNet	**3DmFV**
Batch size	32	64
Point cloud size	1.024	1.024
Optimizer	ADAM	ADAM
Number of epochs	100	100
Number of gaussians	-	125

Benchmarks - Coarse part groups

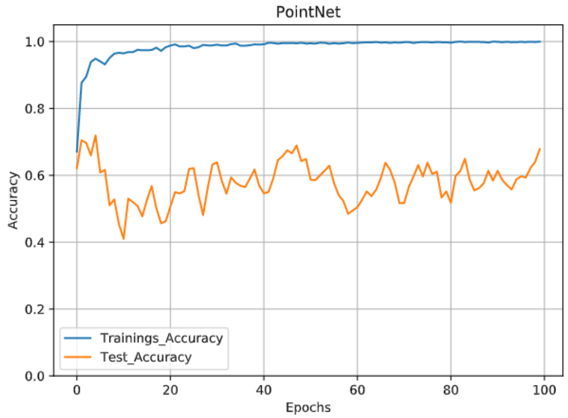
Dresden Database
Systems Group

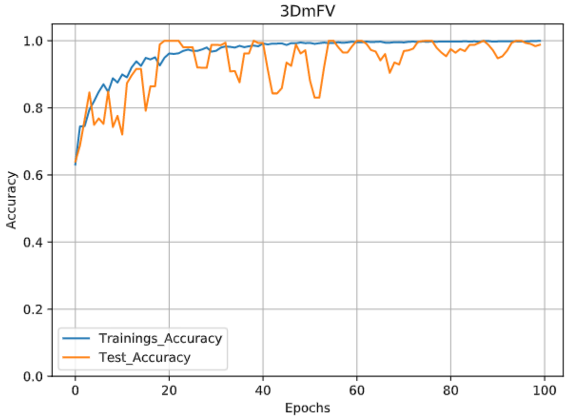
Inner/outer and left/right parts share the same class

- A_pillar and b_pillar
- Overall **15** different classes

Results

- Trainings accuracy:
  - PointNet – 99.5%
  - 3DmFV – **99.7%**
- Test accuracy:
  - PointNet – 71.6%
  - 3DmFV – **98.8%**
- Runtime:
  - PointNet – **16h**
  - 3DmFV – 38h





Benchmarks - Summary

Dresden Database Systems Group

Summary

- Trainings accuracy ~99%
- 3DmFV test accuracy always better than PointNet

→ focus on 3DmFV

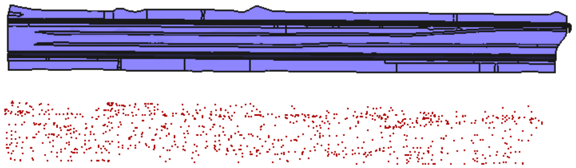
Metric	Approach	Part groups	Distinction left/right	Distinction inner/outer
Classes		15	16	13
Accuracy (Training)	PointNet	99.5%	99.6%	99.8%
	3DmFV	99.7%	99.8%	99.7%
Accuracy (Test)	PointNet	71.6%	68.8%	83.2%
	3DmFV	**98.8%**	**81.0%**	**85.6%**

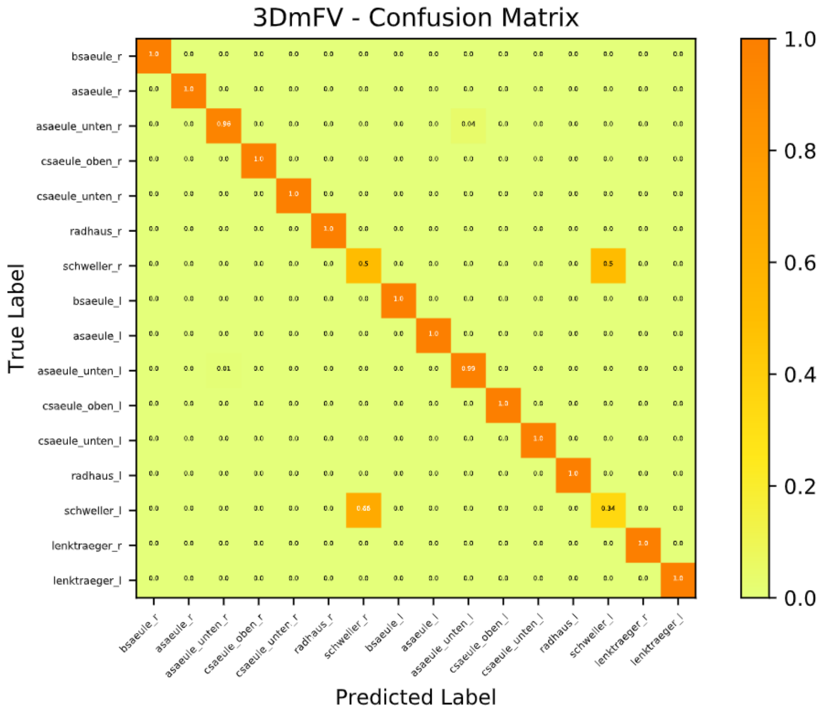
Benchmarks - distinction left/right

Dresden Database Systems Group

Analyze the challenging parts

- Rocker panel seems challenging
- Confusion between left and right part
  → geometric nearly identical

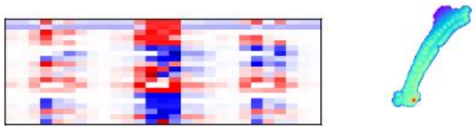


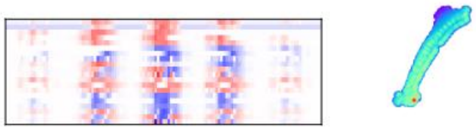


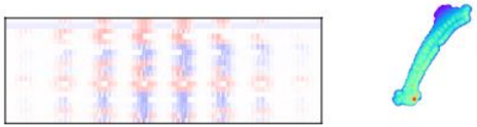
3DmFV - Gaussians

Change number of Gaussians

- Finer grid resolution
- More features
  → better accuracy?
- Distinction between left/right parts


3x3x3


5x5x5


8x8x8

Results

- Slightly better results with finer grid
- Increase in runtime

→ Tradeoff between accuracy / runtime

Metric	3x3x3	5x5x5	8x8x8
Accuracy (Training)	98.5%	99.8%	99.9%
Accuracy (Test)	78.0%	81.0%	**85.0%**
Runtime (Test)	43s	116s	373s

Benchmarks

Dresden Database Systems Group

Conclusion

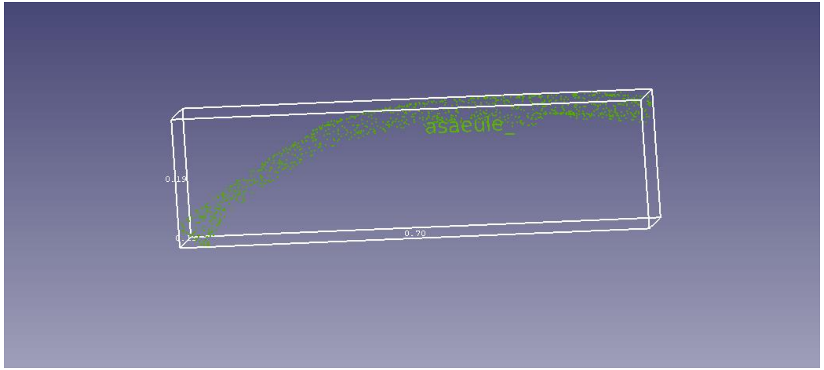
- 3DmFV better performance on tested dataset
- All benchmarks show good results
  → Coarse part groups best one
- Geometric nearly identical parts are challenging
- Tradeoff Accuracy / Runtime

Implementation / Demo

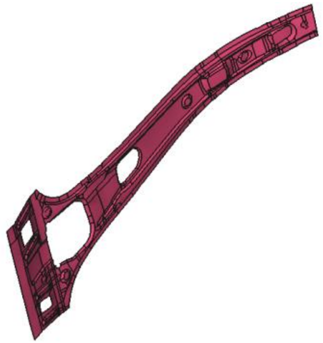
Implementation

Prototype in FreeCAD

- Integrate a trained neural network in a CAD software
- Application for visualizing a Use case
  → Automatic labeling system
- First: Classification of point cloud



→ Classify a part from a STEP file

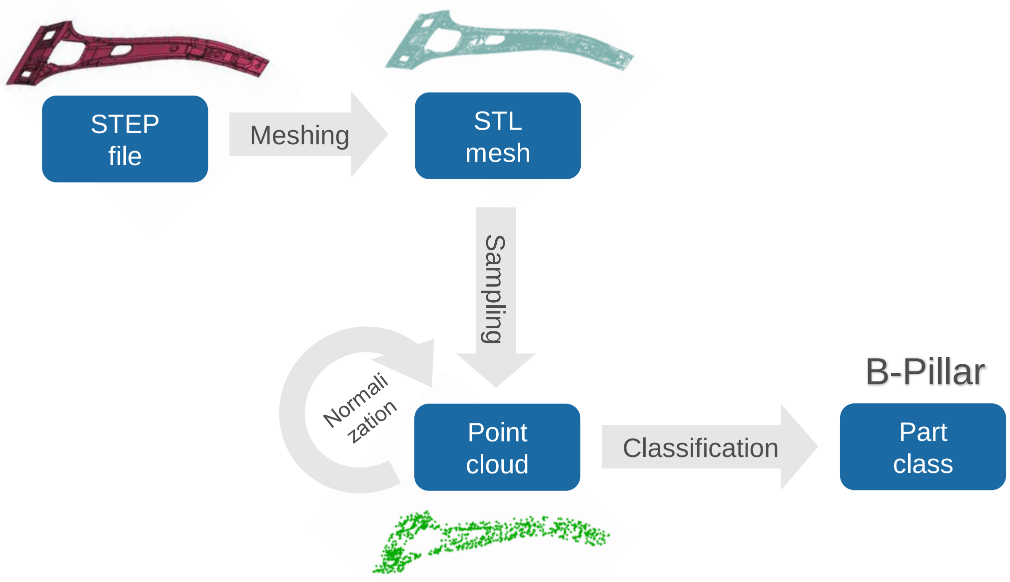


Implementation

Classification pipeline

- Convert STEP into mesh
- Sample mesh with 1024 points
- Normalize points cloud
- Classify point cloud

→ Returns the label of the part and changes the part name



Contribution

Conclusion

- Classification of car parts works!
- 3DmFV shows good results
- Similar parts are more challenging

→ Realization of Use cases are possible!

Outlook

- Training with more parts / models
- Investigate performance of segmentation networks
- Prototype of specific use cases

Thank You!

Discussion

Dresden Database Systems Group

Re: 3DmFV - Question

IS **Itzik Ben Shabat <sitzikbs@gmail.com>**

An: Nick Scheider

Hi Nick,
You are right. it is not rotation or translation invariant. However, this is something that the network learns to make up for. So, the input to the network will be different but the classification will not change despite the rotation and translation. In my case, it was not an issue since the different classes are very different from each other. It may be more challenging if your objects are more similar to one another.
Good luck with your thesis.
BTW, I am no longer using my Technion email. Use this one instead.

Cheers,
Itzik

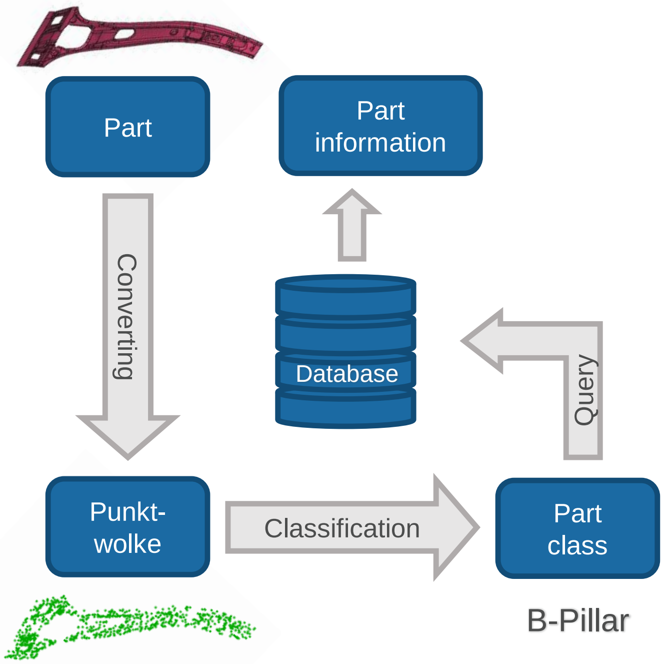
Implementation

Realization of other Use cases

- Collect historical part data in a database
- Use classification network
- Query database with part label

→ return (aggregated) part information

→ Concept of a recommendation system



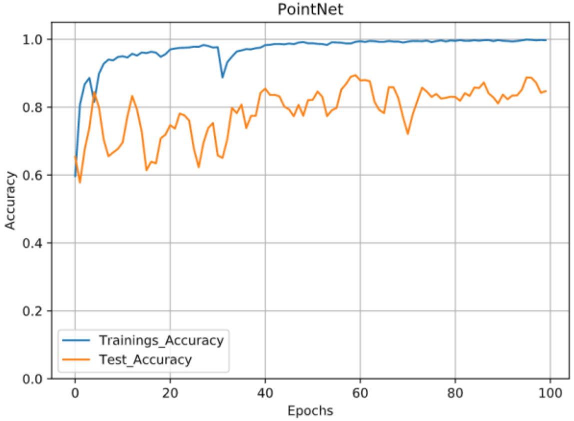
Benchmarks – distinction inner/outer

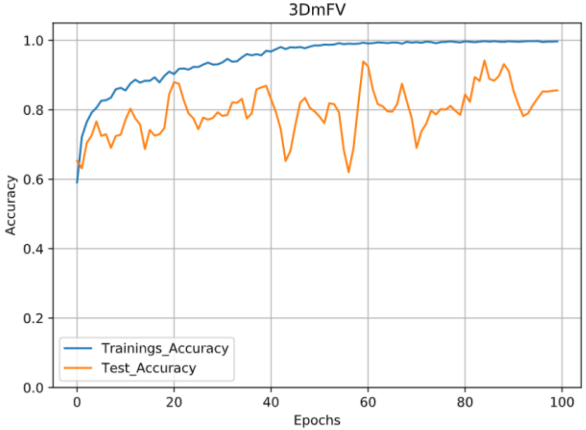
Dresden Database Systems Group

- *Distinction between inner / outer parts*
  - A_pillar_inner and a_pillar_outer
  - Overall **13** different classes

Results:

- Trainings accuracy:
  - PointNet – **99.8%**
  - 3DmFV – 99.7%
- Test accuracy:
  - PointNet – 83.2%
  - 3DmFV – **85.6%**
- Runtime:
  - PointNet – **18h**
  - 3DmFV – 38h



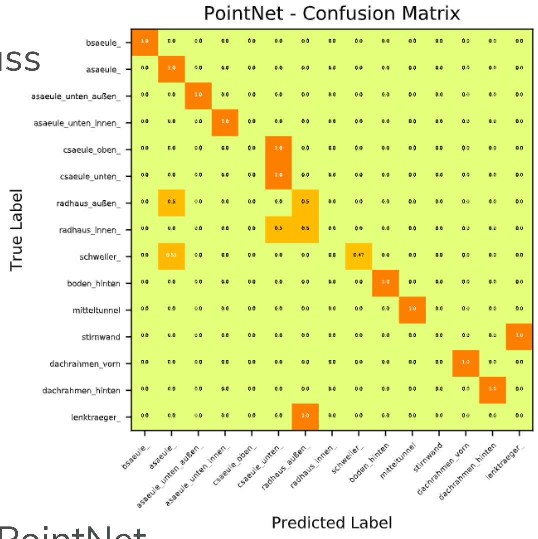


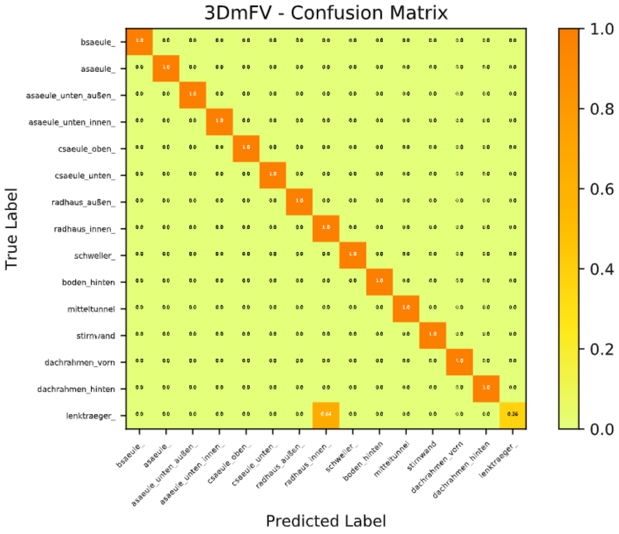
Benchmarks - Coarse part groups

Analyze the challenging parts

- Compare confusion matrix
  → shows the results per class
- Calculate F1 score

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$





Results

- Weighted F1:
  - PointNet – 0.58
  - 3DmFV – **0.96**
- 3DmFV better results than PointNet

Benchmarks – distinction left/right

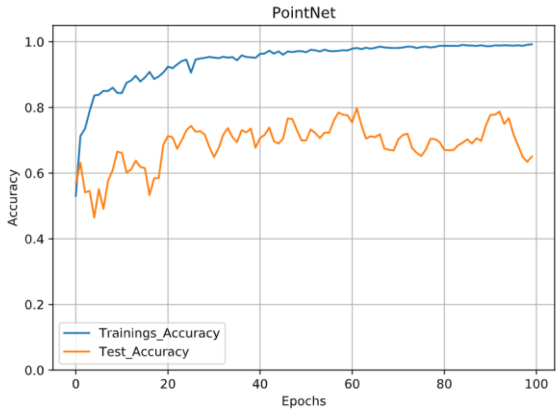
Dresden Database Systems Group

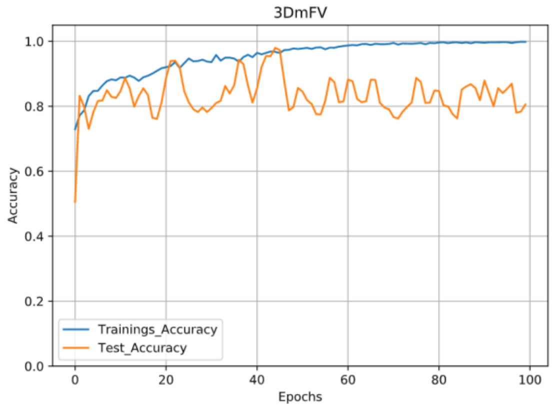
Distinction between left / right parts

- A_pillar_left and a_pillar_right
- Only parts with a counter part
- Overall **16** different classes

Results:

- Trainings accuracy:
  - PointNet – 99.6%
  - 3DmFV – **99.8%**
- Test accuracy:
  - PointNet – 68.8%
  - 3DmFV – **81.0%**
- Runtime:
  - PointNet – **18.5h**
  - 3DmFV – 39h





3DmFV – Different test sets

Dresden Database Systems Group

Test the approach with different test sets

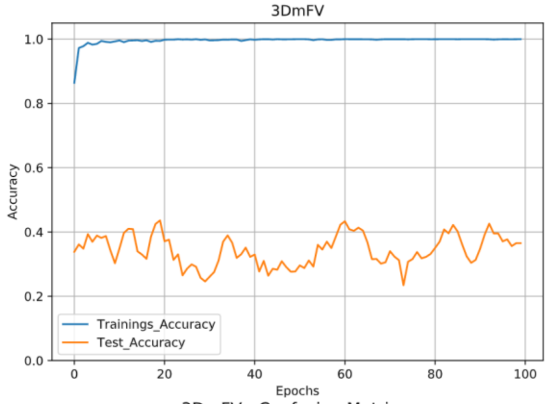
- Only trained on Audi models
- Test with Toyota Yaris dataset
- Classification of coarse part groups

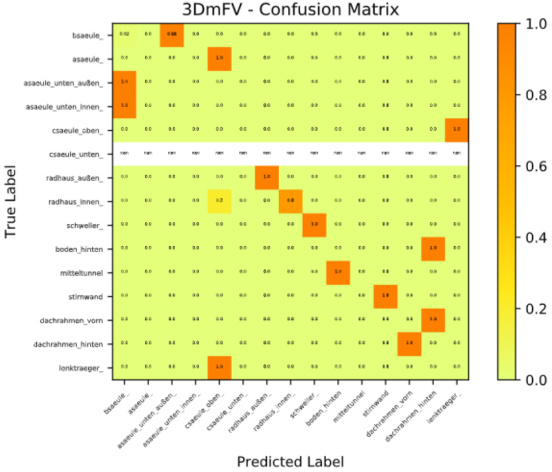
Results

- 3DmFV bad performance
- Test Accuracy: ~38%
- Lots of confusions between classes

Test with Audi FM2

- Same results as before
- Test Accuracy: ~99%





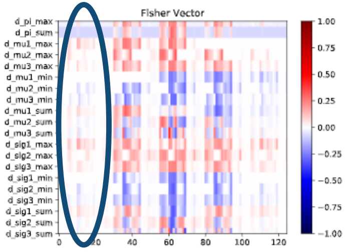
3DmFV – Different test sets

Limits of 3DmFV

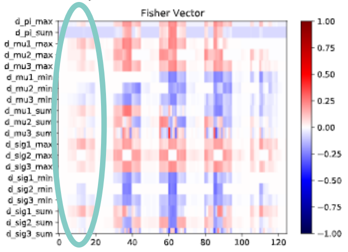
- Comparison of Fisher matrix between B-pillar and a-pillar-lower
- Very similar representation
  → Yaris A-pillar looks more like Audi B-pillar

Conclusion

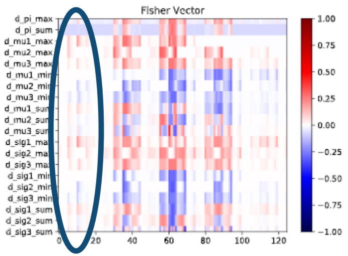
- 3DmFV better performance on tested dataset
- All benchmarks show good results
  - Coarse part groups best one
- Geometric nearly identical parts are challenging
- 3DmFV shows better results on a specific domain – only Audi data



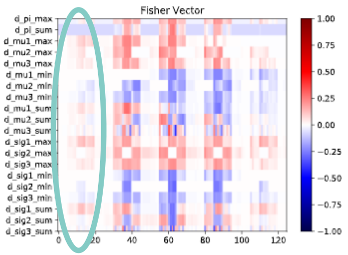
FM2 b-pillar



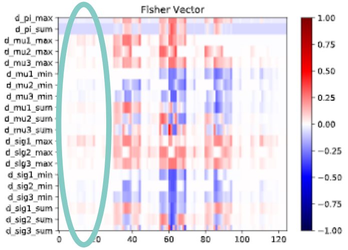
FM2 a-pillar-lower



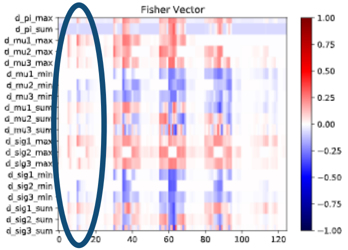
FM3 b-pillar



FM3 a-pillar-lower



Yaris b-pillar



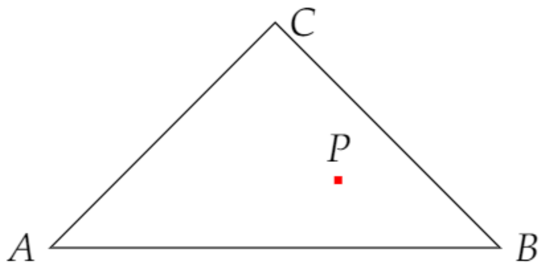
Yaris a-pillar-lower

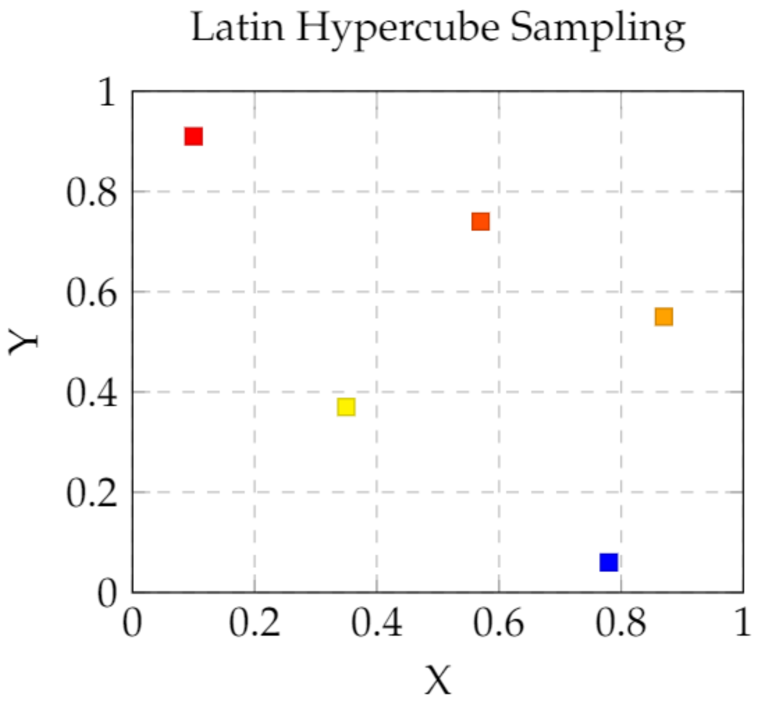
Label Map / Class Map


```
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5     8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 UZ1 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 UZ2 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 UZ3 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 UZ4 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 UZ5 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 WB1 8W0.809.228.B     K  TMZ
bsaeule_innen_rechts, SAEULE_B_INNEN_                - 1.50 - 22MNB5 WB2 8W0.809.228.B     K  TMZ
bsaeule_schließteil_rechts, SCHLIESSTEIL_SAEULE_B_         - 1.00 - HC340XD    8W0.810.222       K  TMZ
asaeule_außen_rechts, SAEULE_A_AUSSEN____OBEN_       - 1.40 - 22MNB5     8W5.810.284.B     K  TMZ
asaeule_innen_rechts, SAULE_A_INNEN_OBEN_            - 1.20 - HC340XD    8W0.809.208.A     K  TMZ
asaeule_unten_außen_rechts, SAEULE_A_UNTEN_____AUSSEN_     - 1.20 - HX340LAD   8W0.809.217.C     K  TMZ
asaeule_unten_innen_rechts, SAEULE_A_IN_UN_____________    - 1.50 - HC450XD    8W0.802.126       K  TMZ
csaeule_oben_rechts, VERST_SAEULE_C_____OBERTEIL_   - 0.75 - HX340LAD   8W5.809.746       K  TMZ
csaeule_unten_rechts, VERST_SAEULE_C_____UNTERTEIL_  - 0.65 - HX260LAD   8W0.809.264       K  TMZ
radhaus_außen_rechts, RADHAUS_HINTEN_____AUSSEN_     - 0.65 - DX56D      8W0.809.412.A     K  TMZ
radhaus_innen_rechts, RADHAUS_HINTEN_IN__________    - 0.85 - DX56D      8W0.810.426.A     K  TMZ
schweller_verst_rechts, SCHWELLERVERSTAERKUNG_NAR__    - 1.50 - HC660X     8W0.809.755.A
schweller_außen_rechts, STEGTEIL_SCHWELLER_            - 0.95 - HC660X     8W0.803.764       K  TMZ
schweller_innen_rechts, SCHWELLER_INNEN____________    - 1.15 - 22MNB5     8W0.803.756       K  TMZ
```

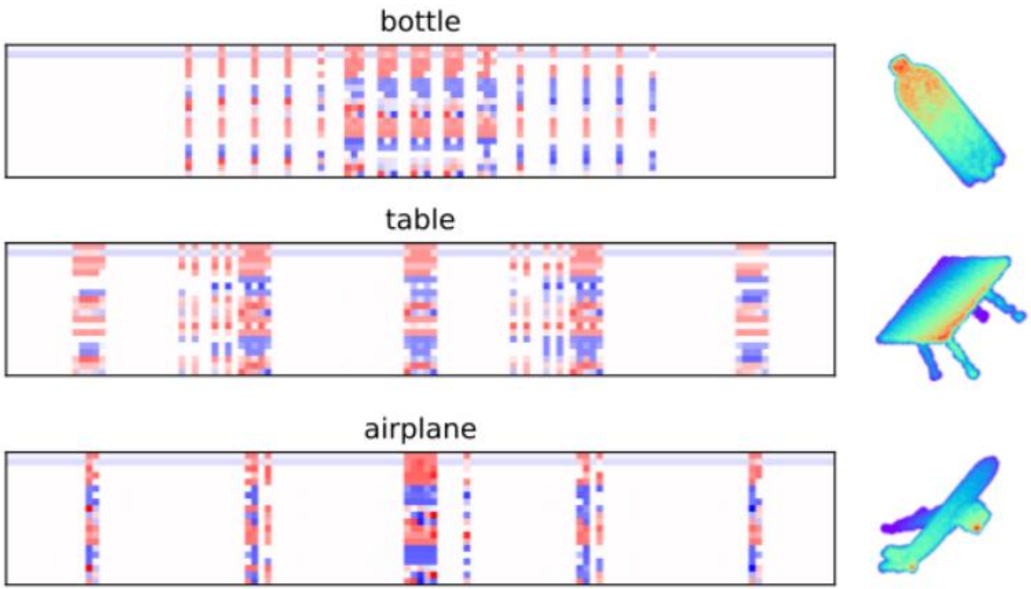
```
bsaeule_innen_rechts,0
bsaeule_schließteil_rechts,0
asaeule_außen_rechts,1
asaeule_innen_rechts,1
asaeule_unten_außen_rechts,2
asaeule_unten_innen_rechts,3
csaeule_oben_rechts,4
csaeule_unten_rechts,5
radhaus_außen_rechts,6
radhaus_innen_rechts,7
schweller_außen_rechts,8
schweller_innen_rechts,8
bsaeule_innen_links,0
bsaeule_schließteil_links,0
asaeule_außen_links,1
asaeule_innen_links,1
asaeule_unten_außen_links,2
asaeule_unten_innen_links,3
csaeule_oben_links,4
csaeule_unten_links,5
radhaus_außen_links,6
radhaus_innen_links,7
schweller_außen_links,8
schweller_innen_links,8
boden_hinten,9
mitteltunnel,10
stirnwand,11
dachrahmen_vorn,12
dachrahmen_hinten,13
lenktraeger_rechts,14
lenktraeger_links,14
```

Sampling



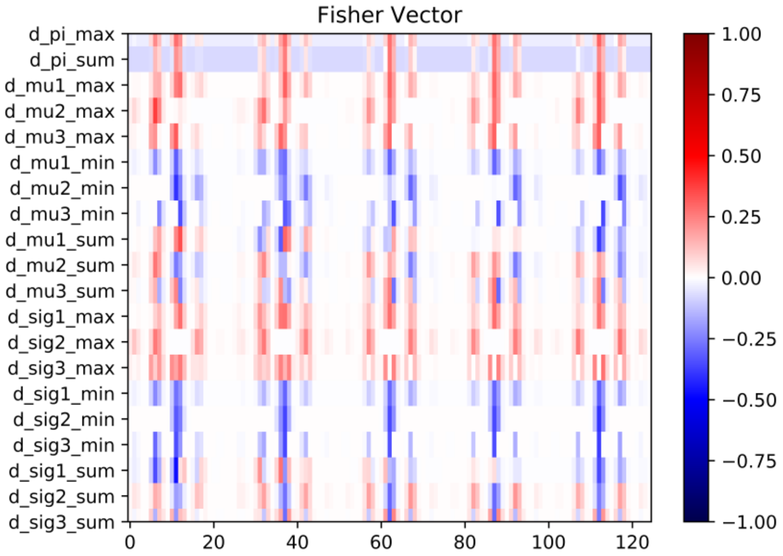


Fisher Vector

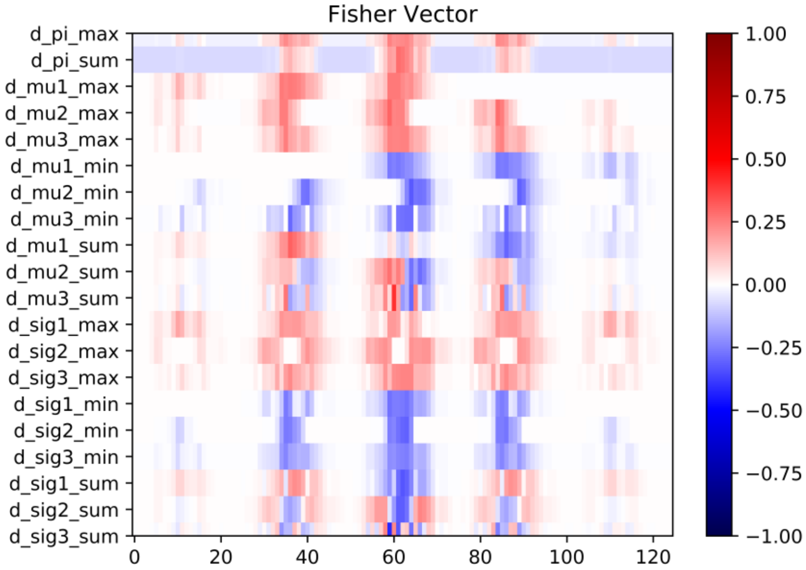


[6]

Fisher Vector



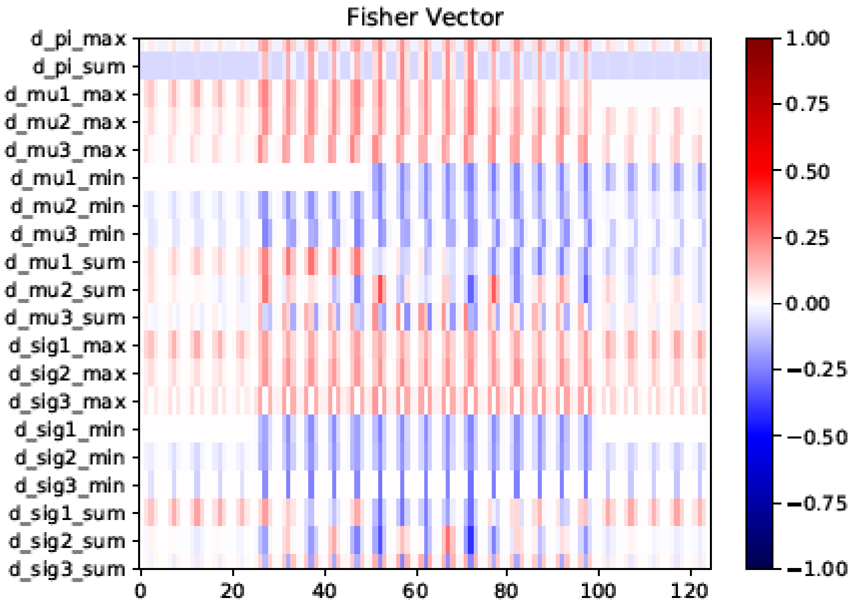
A-pillar



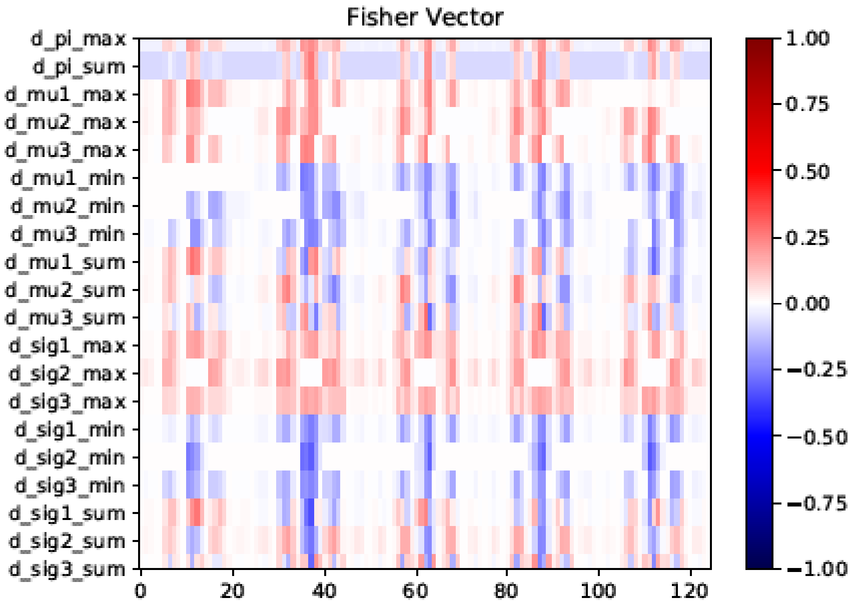
B-pillar

Fisher Vector

Dresden Database Systems Group

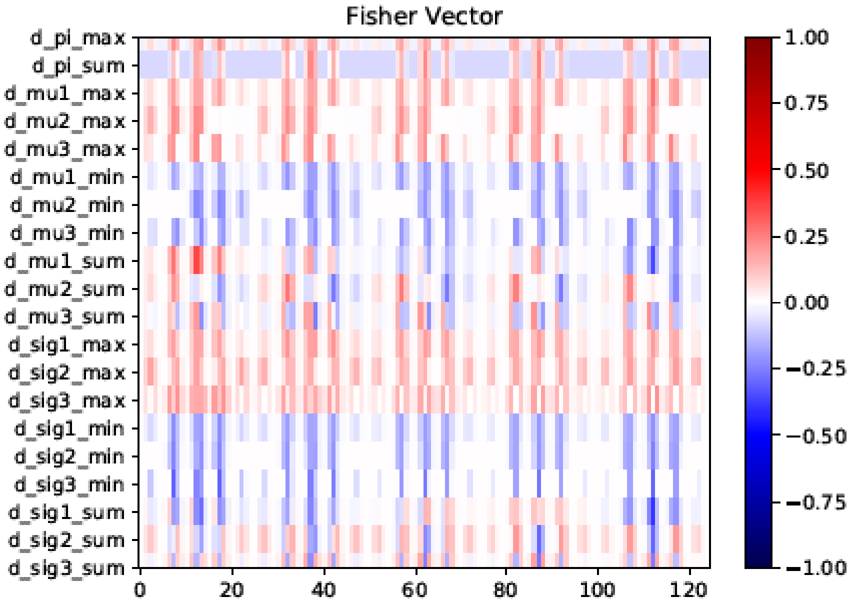


Bottom

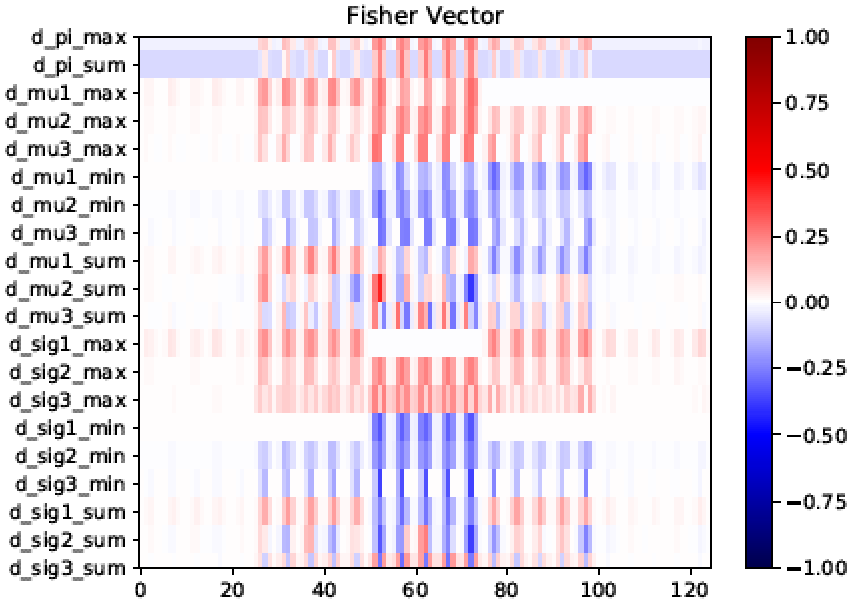


Wheel house

Fisher Vector

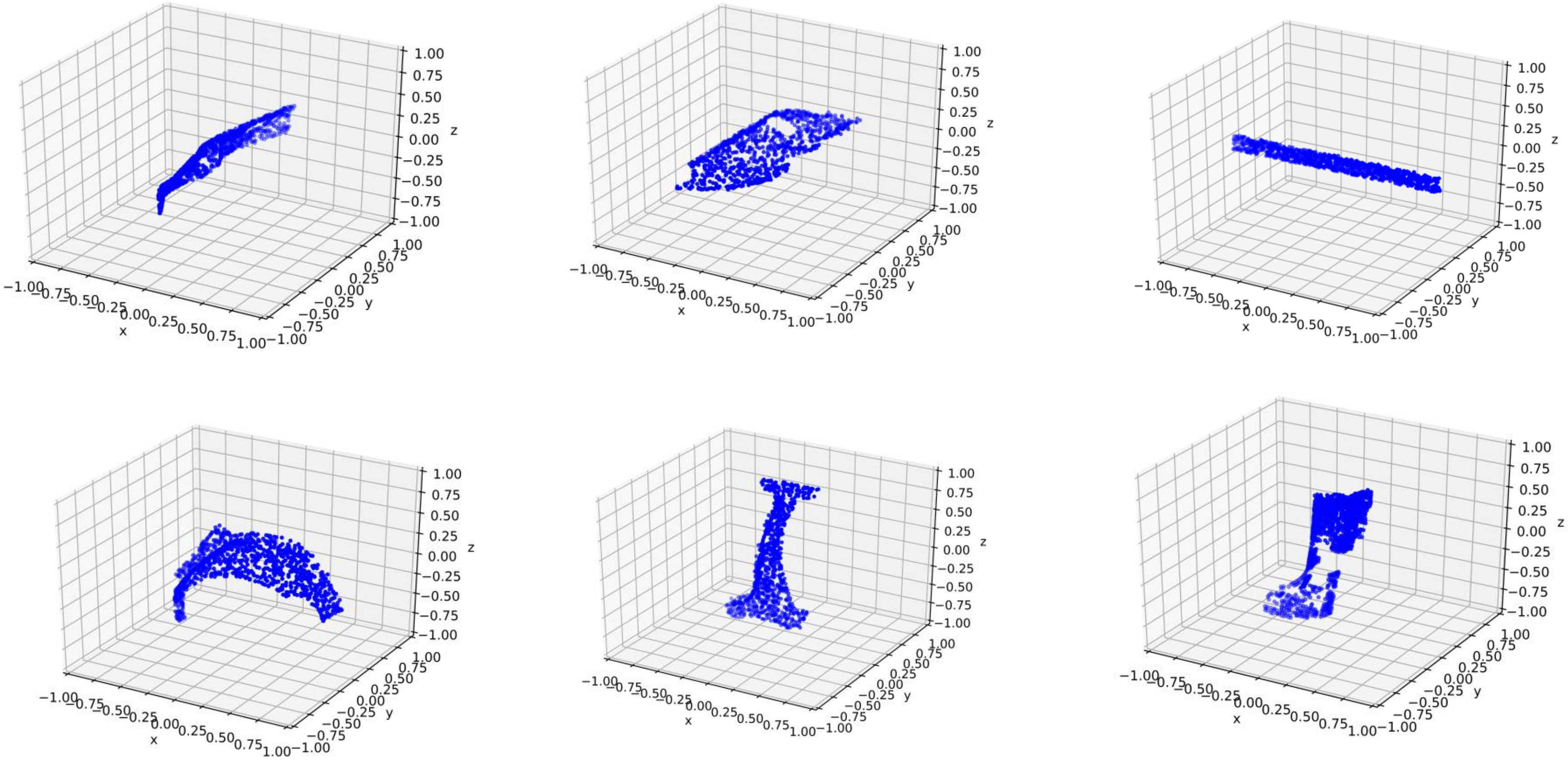


middletunnel



firewall

Parts



Sources

[1] http://www.itzikbs.com/what-is-3d-modified-fisher-vector-3dmfv-representation-for-3d-point-clouds

[2] https://arxiv.org/pdf/1612.00593.pdf

[3] http://www.itzikbs.com/gaussian-mixture-model-gmm-3d-point-cloud-classification-primer

[4] http://www.itzikbs.com/fisher-vector-for-3d-point-clouds-classification-primer

[5] http://www.itzikbs.com/what-is-3d-modified-fisher-vector-3dmfv-representation-for-3d-point-clouds

[6] https://arxiv.org/pdf/1711.08241.pdf